ESTIMATION OF RELATIVE SATELLITE FORMATION ELEMENT POSITIONS IN NEAR CIRCULAR ORBITS

THESIS

Ralph E. Bordner III, Captain, USAF

AFIT/GA/ENY/01M-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GA/ENY/01M-01

ESTIMATION OF RELATIVE SATELLITE FORMATION ELEMENT POSITIONS IN NEAR CIRCULAR ORBITS

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Eductaion and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Astronautical Engineering

Ralph E. Bordner III, B.S.AsE.

Captain, USAF

March 2001

AFIT/GA/ENY/01M-01

ESTIMATION OF RELATIVE SATELLITE FORMATION ELEMENT POSITIONS IN NEAR CIRCULAR ORBITS

Ralph E. Bordner III, B.S.AsE.

Captain, USAF

Approved:

William E. Wiesel	7 Mar 01
Dr. William E. Wiesel Jr. Thesis Advisor	Date
Steven G. Tragesser	06 MAR 01
Dr. Steven G. Tragesser Committee Member	Date
David R. Jacques	6 MAR 01
Lt Col David R. Jacques Committee Member	Date

Acknowledgements

This thesis could not have been accomplished without the support from several incredibly talented and insightful people. First, I would like to express my sincerest thanks to my advisor, Dr. William Wiesel. I would like to publicly say thank you for all of the time you took to advise me both in your office and through endless emails. Additionally, thanks also need to go to my thesis committee, Lt Col David Jacques and Dr. Steve Tragesser, for their thoughts and recommendations. I also received assistance outside the core individuals listed above. First and foremost, thank you Mike Violet for all that you did to help me in this effort and for your mentoring. Only a true friend could have read this thesis (especially in its infancy) so many times. Also, thank you for reminding me during the tough times that with much perseverance, even the snail made it to the Ark. I would also like to thank Major John Raquet, Captain David Irvin, and Captain Mike Novy for the roles that they played (mostly listening to me ramble). On a more personal note, I would like to thank God, my wife and my baby daughter for helping to keep me
focused and for their understanding during the times when I was either glued to my PC or hunched over at the table doing endless homework. Without their patience (not to mention reminders to eat and bathe), I could not have finished this work or completed this degree. Thank you so much for putting up with a grumpy
and extremely busy husband for so long! Te Amo! Finally, an extra special thanks goes to my baby girl whose selfless and toothless smile reminded me on a daily basis what life is really about.

Ralph E. Bordner III

Table of Contents

List of Figures

List of Tables

Abstract

Precise relative position determination of satellite formation elements in near circular orbits is the focus of this thesis. A Bayes filter is used for the estimator with the dynamics solution obtained by solving the time periodic, linearized system via Floquet Theory. This approach is similar to that of the traditional Clohessy-Wiltshire treatment, however much more of the system's behavior is retained by using a time periodic linear system. The dynamics model includes all zonal harmonics of the earth as well as sectoral, tesseral, and air drag perturbations. The filter algorithm was given three types of simulated measurements: relative Carrier-Phase Differential GPS data, Standard Positioning System GPS data and range measurements between each satellite. Simulations were conducted to investigate the accuracy obtainable with these measurements. Results obtained are on the order of the noise of the measurements (as low as 2 centimeters in each coordinate axis).

ESTIMATION OF RELATIVE SATELLITE FORMATION ELEMENT POSITIONS IN NEAR CIRCULAR ORBITS

I. Introduction

1.1 Background Information

The path to understanding any subject usually begins with the introduction of fundamental terms and their definitions. The process of understanding more recent space system architectures is no exception to this trend, thus this work will begin with a fundamentally oriented approach as well. Technological advances over the last few years have increased the complexity and size of current space system architectures. One type of architecture that has received increasing attention is one which has a distributed nature. This type of architecture has three important terms that must be properly delineated from each other to allow proper understanding of the subject. These terms are a satellite cluster, a satellite formation, and a distributed satellite system. These terms are often used interchangeably as though they were synonyms; however, as they are described in a paper by Shaw [27], their definitions do differ. The following text shall illustrate this difference.

A satellite cluster is simply a group of two or more satellites located in different, yet possibly similar, orbits performing a desired mission for a local area as if it were one large satellite. This work will refer to the individual satellites within a satellite cluster as elements, and the cluster's particular application will drive the positioning of these elements. In general, they are positioned in one of two ways. First, the application of the cluster may require no special geometry of the formation, only that the cluster elements be distributed. This is usually found when a system

is designed with more satellites than are required to accomplish its mission. In other words, two or more satellites service the same locality and the mission could be accomplished with less than the total number. The rationale behind this type of cluster is to provide sufficient redundancy so that mission performance will not experience significant degradation during an individual element's failure. The second paradigm to cluster design is the more common positioning scheme. This scheme is used when the cluster geometry must have a specific shape and that the relative positions of the spacecraft to each other be known with some degree of accuracy, sometimes even to exquisite levels. This type of satellite cluster is referred to as a formation since the cluster geometry is designed with purpose and operational steps are taken to maintain that geometry. Sensing applications, such as space-based radar, would use a formation instead of a cluster as the elements would need to coordinate amongst themselves to form a single aperture. Formations may use active or passive techniques to control the shape of the formation. The decision of what control method to employ depends on the application and how important the formation geometry plays in mission execution. For example, the formation's mission may drive operations to explicitly maintain not only relative positions but a specific orientation as well. This would imply active control methods such as thrusters. Unfortunately, this type of formation drives control costs to unrealistic levels if the mission duration of the cluster is of any significant length. A more realistic approach is to reduce control costs by allowing the elements to follow trajectories about a reference orbit. While the resultant motion of the spacecraft will cause the orientation of the formation to change (and possibly the relative distances between the spacecraft), the trajectories chosen will maintain the elements within a predetermined envelope of space. Thus, the formation will remain together with less maneuvering costs. The concept of satellite formation flying has been a topic of considerable study for well over a decade. In fact, P. S. Visher was given a patent for the concept in 1983 [29]. At this point, it is important to note that a cluster or formation may be a subset of a

larger architecture called a distributed satellite system. A distributed satellite system is any space system composed of many satellites designed to coordinate as one to accomplish its mission for a large region, or at least larger than one locality. This may sound identical in its definition to a satellite cluster, but the difference is that a satellite cluster services a *local* area while a distributed satellite system services a *global*, or larger, area.

The reasons for using a satellite cluster or formation, as opposed to a single larger satellite, include providing on-orbit redundancy, reliability, survivability, reducing cost and improved performance (or possibly enabling performance that cannot be accomplished via a single satellite). An example demonstrates these gains easily. Imagine a satellite cluster comprised of ten satellites. Each satellite within the cluster contributes to the overall objective of the system. If one were to fail, the integrity of the system is not totally lost as the remaining nine can still function to achieve mission objectives. While the overall capability of the system may be temporarily diminished, it is not completely lost. Also, if the cluster is operating with more than the required number of elements to perform its objectives, the mission may not experience any degradation to its minimum advertised capability due to the on-orbit redundancy. A related concept to reliability is survivability. The above example illustrated continued system performance during a failure, and thus improved reliability. Survivability also implies continued performance but after the infliction of damage due to adversaries or on-orbit mishaps. Both scenarios are very similar as failure is involved in either case. However, the latter case is much more of a concern for DoD users, specifically the military. Military systems could improve their survivability by either moving current surface, or near-surface missions, to space or distributing current space-based systems into clusters or formations.

While the advantages of reliability, survivability, and redundancy are appealing, the biggest draw to the idea of satellite clusters may be the gain in performance or the enabling of performance not otherwise possible. In applications such

as radar, performance that could not normally be accomplished by one satellite can be achieved with a distributed system. In particular, potential improvements to ground moving target indication (GMTI) radar systems are very attractive since the resolution of such a space-based system has the potential to be much greater than current systems. This improvement can be made by distributing a system to form a synthetic aperture. By enlarging the size of the cluster, and thus increasing the size of the aperture, the resolution increases. If the radar wavelength, λ, and the range, R, to the target is assumed constant, the ground resolution, r, of the radar is inversely proportional to the size of the synthetic aperture [16].

$$r = \frac{R\lambda}{2S} \tag{1}$$

Presently, the United States Air Force (USAF) employs the Joint Surveillance and Targeting Attack Radar System (JSTARS) within regions of military interest to detect ground targets. This airborne mission could be moved to the space arena for potentially dramatic improvement in ground resolution. A space-based version of JSTARS would not only enhance mission performance but also reduce the system's vulnerability to attack. This topic has been looked at from various angles, including signal processing, detection probability, and orbit design [26, 15, 11].

While observing missions seem to be the common theme, space-based radar is not the only application being studied. The following represent just a few of the ideas currently under review. The European space agency has developed an elaborate formation for its cluster mission to study the Earth's magnetosphere. The Orion project, funded by NASA Goddard Space Flight Center, is a low-cost demonstration of GPS uses in formation flying. The laser interferometer space antenna (LISA) mission is a heliocentric formation flying mission designed to detect gravity waves. Also, the Air Force Research Laboratory's (AFRL) TechSat21 program is a satellite formation that will act as a virtual satellite. Among its functions will be that of space-based radar [23].

1.2 Satellite Formation Flying Issues

Several technical issues must be overcome in order to enjoy the advantages of satellite formation flying. When one thinks of problems that would arise from a system of this sort, the first thing that comes to mind is keeping them together. The methods to overcome this problem range from tethers to precision sensor and actuator systems using state of the art inertial sensors or Global Positioning System devices to maintain formation. Tethers may not require active control to maintain formation, however the dynamics behind keeping the system stable are extremely complicated to say the least. Active methods using sensors and actuators are usually comprised of thruster firings to perform stationkeeping. Active methods also have dynamics issues that have to be addressed as well as precise estimation of element positions and control schemes. If the mission duration is short enough or the positioning requirements loose enough (or both), a designer may opt for neither of the above solutions. A more simple (and cheaper) passive method is to choose the initial conditions of the formation such that the formation experiences limited secular effects. Thus, the formation will feel as little of the effects that cause dispersal. This was the control method chosen by the Magnetospheric Multiscale Mission (MMS). They chose slight differences in two of the classical orbital elements, specifically the right ascension of the ascending node (RAAN) and the inclination, to allow the formation to maintain itself over its mission duration [18]. A few stationkeeping maneuvers are expected but they are few and relatively small. While the method to control the relative positions of the elements and the orientation of the formation geometry is important, it is not the only issue that faces formation flying. The major issues include cooperative control, formation geometry design, orbit determination, and getting the system synchronized to act as one.

The work of Sabol et al [23] investigated four different types of formation geometry. These are an in-plane formation, an in-track formation, a circular formation, and a projected circular formation. The in-plane formation was the simplest

of the formations they studied. It consisted of several satellites populating the same orbital plane but separated by mean anomaly. The in-track formation is a slight variation of the in-plane in that the satellites share the same ground track. This was accomplished by varying the orbital planes by right ascension of the ascending node. The last two maintain the geometry described in their name with the difference to the latter being that this cluster only maintains a fixed distance in the along-track/cross-track plane. Sabol's results showed that the instability of each geometric design increased from in-plane to circular (the order in which they were introduced above) as one might assume. The study paints a grim picture for control costs. A worst-case scenario has estimates totaling near 50 meters per second annually to mitigate the effects of perturbations on the designed orbits.

Control cost is a leading, if not the leading, challenge facing the implementation of formation flying. If control costs cannot be lowered to more realistic levels, the money and effort required to make satellite formations a workable reality will outweigh any advantages gained by their use. Simply put, the concept will not be practical. Andrew Sparks of the AFRL has written several papers on this topic, and one in particular [28] shows results consistent with Sabol et al. However, he does show control costs as low as .022 meters per second per day (near 20 meters per second per year) by varying the orbital parameters in his models. Regardless, 20 meters per second per year is still an impractical fuel requirement. Control costs this large translate to short lifespans and increased budgets which argue against the feasibility of satellite formations.

Research into the position estimation of the elements within a formation, either relative or absolute, has received little attention. However, this area of research is an important one as many formation flying missions require that the relative positions of each element with respect to the others be known with at least some degree of accuracy. The degree of accuracy depends on the mission. Space-based radar applications require positions to be known to one tenth of the radar's wavelength [3].

Several Masters theses at the Air Force Institute of Technology have been focused on the topic of position estimation[29, 12, 9]. Most of them used a U-D covariance factorization Kalman filter that operated on range data exchanged between the satellites. The relative position error between the satellites estimated by this filter was on the order of 3 cm. A more recent study by Captain Jeffery Davis [8] using a Bayesian approach attempted to estimate relative satellite positions by using absolute data. While the research did not have favorable results, it does serve as the starting point and a source of lessons learned for this research.

1.3 Problem Description/Objectives

All of the issues discussed previously are integral to a successful system design, however this work will focus on the estimation of the relative positions of the elements within a formation. While determination of the relative positioning of the spacecraft is the primary objective of this work, some consideration will also be given to determining the absolute positions. The ideal result of this thesis would be to create an estimator that has extremely precise relative positioning performance (on the order of centimeters) and adequate absolute positioning (on the order of tens of meters). While absolute positional information is necessary to establish communications with the ground segment and for determining field of view, its accuracy requirement is more lax than the relative requirement for space-based applications such as radar or earth sensing missions. Absolute positioning knowledge with error in the hundreds of meters may be acceptable in most cases since the area covered by the ground antenna's uplink radiation at orbital altitudes can be on the on the order of kilometers.

Recursive estimation techniques will be used in this thesis to determine if relative range information from satellite timing signals, Global Positioning System (GPS) data and Carrier Phase Differential GPS data will be sufficient to calculate relative satellite positions to the tight accuracy requirements previously described.

As alluded to earlier, this thesis is a continuation of work done by several AFIT students, particularly that of Captain Jeffery Davis. However, while Captain Davis investigated this problem from the standpoint of determining relative element positions using absolute data, this work does so using only relative data. It is important to stress that the previous attempt using only absolute data was not accomplished with favorable results since the filter ran out of significant digits (using a double precision computer) causing covariance inversion problems. This work uses only relative data to try to separate the relative positioning from the absolute positioning problem to avoid this situation. Some of the measurements used were absolute in nature (i.e. GPS measurements), however this data was manipulated into a relative form prior to use. More details and insights into the particulars of the data used are provided in the Methodology section.

Arguably, the relative dynamics model is the most important part of a satellite formation's on-orbit estimator. Any inaccuracies in the dynamics model to predict the behavior of the formation will be returned in the form of perceived erratic behavior in the motion of the elements of the formation by the filtering algorithm. On-board control mechanisms will then act upon this perceived erratic motion, resulting in the excessive use of stationkeeping fuel. This action would be especially unfortunate as the erratic motion predicted may be nothing but harmless periodic motion. If left uncorrected this motion would show the satellites moving within a bounded envelope of space. With proper modeling to predict the movement of each spacecraft, this motion would be shown to not degrade the spacecrafts' mission and thus excessive maneuvering could be eliminated.

In order to meet the stringent relative positioning accuracy requirements inherent in space-based radar applications, this thesis uses work accomplished by Dr. William Wiesel [36, 35] in the area of relative satellite motion as the heart of its estimator's dynamics model. Dr. Wiesel is a Professor of Astronautical Engineering at the Air Force Institute of Technology. His relative motion work involves a new

approach to relative satellite motion about an oblate planet by linearizing about a nearly circular periodic orbit as a reference solution. By using Floquet theory to solve the linear system, a first order relative motion solution was produced that included all zonal harmonics of the earth's gravitation field, gravitational harmonics, and air drag. Analysis shows that this new solution, while similar to the classic Clohessy-Wiltshire solution for relative satellite motion, performs three orders of magnitude better [36]. This author expects this relative dynamics solution will provide sufficient insight into relative satellite dynamics such that satellite formation navigation will be improved for space-based radar applications.

II. Background

2.1 Relative Motion

Relative spacecraft motion has long been a problem for mission analysts who plan rendezvous maneuvers. These planners look to the solution devised by Clohessy and Wiltshire as their primary analysis tool. These two individuals reduced the problem of relative satellite motion to the familiar two body problem by assuming circular orbits, linearizing the system, and then solving the resultant equations of motion. This problem was first looked at by Hill in 1878, but it was revisited by Clohessy and Wiltshire in the 1960's and their work made it practical for engineers [30]. While the original intent of these equations was to describe the relative motion of one closing spacecraft with respect to a target spacecraft, the Clohessy-Wiltshire (CW) solution has been the starting point of analysis for satellite formation dynamics.

2.1.1 Clohessy and Wiltshire. Although the CW solution is not used in this work, a brief description is warranted to highlight the reasons why it was replaced with a different, yet analogous, solution in this thesis's filter. As mentioned previously, the CW equations are the solution to the linearized circular orbit problem and in its simplest form, no perturbations are modeled. The differential form of the CW equations, for an earth based scenario, are

$$\ddot{x} - 2n\dot{y} - 3n^2x = 0, \tag{2}$$

$$\ddot{y} + 2n\dot{x} = 0, \tag{3}$$

$$\ddot{z} + n^2z = 0, \tag{4}$$

where n is the mean motion

$$n = \sqrt{\frac{\mu_\oplus}{R^3_{radius}}} \tag{5}$$

and R_{radius} is the distance from the center of the earth to the target vehicle in circular orbit [30]. The standard reference frame is a radial, orbit normal, and in-track frame (see Figure 1). Any perturbations to be included within this set of equations would

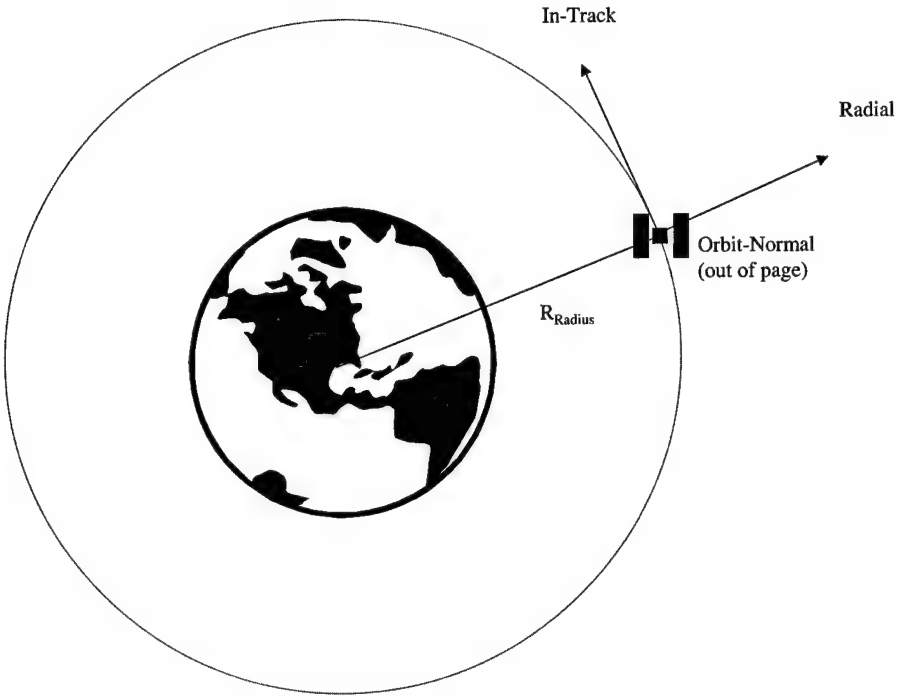

Figure 1 Clohessy-Wiltshire Reference Frame

be modeled as acceleration terms on the right hand side of equations 2, 3 and 4. Since these equations are set to zero (i.e. no non-central force effects), the resultant motion can be described as free motion. These equations can either be solved via the standard eigenvalue-eigenvector approach or through a more simpler approach if one utilizes simple harmonic motion [31]. The solution can be found in texts by Wie [30] and Wiesel [31] as well as most astronautical engineering textbooks.

Under this free motion, the satellites within a formation will follow 2 by 1 ellipses along the trajectory of the orbit of the satellite and in the vertical plane of motion [25]. This motion is periodic and thus while the relative distances between the satellites will change, the size of the cluster will be bounded. The work of

Sedwick showed that secular and periodic perturbations are considerable forces that destroy the simplicity of the free motion model. The key to effective modeling, and thus efficient stationkeeping, is to account for all forces that may cause the orbital positions to diverge, leaving only benign forces that will cause the satellites to move together or move periodically about.

Even with the inclusion of significant perturbation acceleration estimates, the results of the CW solution will probably not be optimistic for the navigation of a satellite formation. The approximations made in Hill's equations, namely using a time-invariant system, make the dynamics the weak link in most of the work done concerning satellite formations. Also, the CW solution is being used for purposes other than its original intent. Thus, improvement in the relative dynamics model is key in the effort to validate whether the idea of satellite formation flying is feasible.

2.1.2 Wiesel. Dr. Wiesel's work takes the general idea of the CW solution and extends it to include much more of the true dynamics of the system. Rather than linearize about a circular orbit, he applies Floquet theory to linearize about a near circular, *periodic* orbit. The advantage of integrating a periodic orbit over a single, particular orbit is that more of the behavior of the orbit is understood as time approaches infinity [32]. This is due to the fact that the orbit motion is simply a repeat of itself every period. The result is a linear, time periodic system instead of a linear, constant coefficient system. His work shows two secular terms in its solution. It is noteworthy to mention that this appearance of two secular terms is in agreement with findings of Schaub and Alfriend and an improvement over the usual one secular term discovered due to the earth's oblateness [36, 24]. Thus, it appears that this new solution is an improvement over the classic CW solution. Although most of this improvement can be attributed to the shift from a time-invariant system to that of a time periodic system, the dynamics solution includes many perturbation sources. Dr. Wiesel's dynamics model was created such that all major perturbative forces could be modeled to better predict satellite behavior and minimize fuel usage. In

summary, this new approach holds three advantages. First, its solution inherently retains *more* information about the orbit over time leading to the introduction of *less* error into the dynamics. Second, it reflects two secular modes. Since both secular modes are modeled by the dynamics (rather than the usual one), they each can be identified and compensated for without unnecessarily beating them down with stationkeeping maneuvers. And finally, this solution allows for the inclusion of a large number of the earth's zonal harmonics (this thesis models zonal effects through the 14th order), gravitational harmonics, and air drag, achieving results closer to reality through extensive modeling of perturbations. The following text will overview the theory in Dr. Wiesel's paper [36], but by no means is an exhaustive treatment. The reader is encouraged to peruse the paper for full details.

Dr. Wiesel conducts his work from a Hamiltonian perspective. It begins with identifying the Hamiltonian function pertaining to the orbital motion of a small body about the earth. By using dimensionless units with radius of the earth $R_\oplus = 1$, the gravitational constant G=1, and the mass of the earth $M_\oplus = 1$, the Hamiltonian function in the inertial frame is found to be

$$\begin{aligned} \mathcal{H} &= \frac{1}{2}\{P_x^2 + P_y^2 + P_z^2\} - \frac{1}{r} \\ &+ \frac{J_2}{2r^3}\left\{3\frac{Z^2}{r^2} - 1\right\} + \frac{J_3}{2r^4}\left\{5\frac{Z^3}{r^3} - 3\frac{Z}{r}\right\} \\ &+ \frac{J_4}{8r^5}\left\{35\frac{Z^4}{r^4} - 30\frac{Z^2}{r^2} + 3\right\} + \ldots \end{aligned} \tag{6}$$

The inertial state vector is defined as $\mathcal{I}^T = (X, Y, Z, P_x, P_y, P_z)$, thus the radius, r, is defined as $r = \sqrt{X^2 + Y^2 + Z^2}$. The momenta states, (P_i), are just the inertial velocity components since the Hamiltonian is in terms of mass per unit satellite. Notice, there are no other perturbations included in this Hamiltonian function. One of the advantages of Hamiltonian dynamics is the ease of adding perturbations. Additional conservative perturbations can be modeled by the addition of the potential function of the perturbating force (i.e. sectoral, tesseral, etc.). Nonpotential forces (i.e. air

drag) are expanded about the periodic orbit, and then added to the Hamiltonian equations of motion for the momenta states [36].

Since the orbit in this system does not return to the same position in inertial space, a rotation to the nodal frame (where the orbit closes on itself or is periodic) is performed to compensate for the regression of the orbit along the equator. The resulting Hamiltonian is found to be

$$\begin{aligned} \mathcal{H}' &= \frac{1}{2}\{P_x'^2 + P_y'^2 + P_z'^2\} \\ &+ \dot{\Omega}(P_x'Y' - P_y'X') + V(Z', r), \end{aligned} \tag{7}$$

where the state vector components marked with a $'$ are just the inertial coordinates rotated into the nodal frame, $\mathcal{N}$. The state vector in the nodal frame is now defined by $\mathcal{N}^T = (X', Y', Z', P_x', P_y', P_z')$. The $V(Z', r)$ term represents the conservative zonal harmonic terms rotated into the nodal frame. This nodal frame is not the only frame in which the system is periodic. The system can be moved to the classic CW frame and still retain its periodic quality. To do this, first the periodic orbit must be transformed to the origin of the current coordinates. This is done by subtracting off the periodic orbit from the state vector in the nodal frame, thus the new local coordinates are given by

$$x = X' - X'_o, \tag{8}$$

$$p_x = P'_{xo}(t), \tag{9}$$

$$y = Y' - Y'_o, \tag{10}$$

$$p_y = P'_{yo}(t), \tag{11}$$

$$z = Z' - Z'_o, \tag{12}$$

$$p_z = P'_{zo}(t), \tag{13}$$

where the quantities subscripted with o correspond to the periodic orbit. This reduces the Hamiltonian to its second order and higher terms evaluated on the periodic orbit. Thus, with the new local coordinate vector defined as $\mathcal{L}^T = (x, y, z, p_x, p_y, p_z)$, the new Hamiltonian becomes

$$\mathcal{H}'' = \frac{1}{2!}\mathcal{H}''_{\alpha\beta}\mathcal{L}_\alpha\mathcal{L}_\beta + \frac{1}{3!}\mathcal{H}''_{\alpha\beta\gamma}\mathcal{L}_\alpha\mathcal{L}_\beta\mathcal{L}_\gamma + \dots \tag{14}$$

The $\mathcal{H}''$ terms are fully symmetric partial derivative tensors evaluated on the periodic orbit. In other words,

$$\mathcal{H}''_{\alpha\beta} = \left.\frac{\partial^2\mathcal{H}'}{\partial\mathcal{N}_\alpha\partial\mathcal{N}_\beta}\right|_o \tag{15}$$

and

$$\mathcal{H}''_{\alpha\beta\gamma} = \left.\frac{\partial^3\mathcal{H}'}{\partial\mathcal{N}_\alpha\partial\mathcal{N}_\beta\mathcal{N}_\gamma}\right|_o, \tag{16}$$

where repeated Greek indices are summed from one to six.

Now, a final rotation is accomplished so that the solution will be in the familiar CW frame (radial, in-track, cross-track). Figure 1 shows this frame pictorially. This thesis requires the generation of a linear system for its estimator, thus the Hamiltonian is truncated after the second order terms. The final form of the Hamiltonian, $\mathcal{K}$, to the second order in the CW frame is

$$\begin{aligned}\mathcal{K}_2 &= \frac{1}{2}\mathcal{H}''_{\alpha\beta}\mathcal{R}^{(2)}_{\alpha\gamma}\mathcal{R}^{(2)}_{\beta\delta}\mathcal{Z}_\gamma\mathcal{Z}_\delta \\ &+ \mathbf{p}\cdot\dot{\mathcal{R}}\mathcal{R}^T\mathbf{q},\end{aligned} \tag{17}$$

where the $\mathcal{H}$ term is a fully symmetric partial derivative tensor evaluated on the periodic orbit, the $\mathcal{R}^{(2)}$ terms are six by six block diagonal matrices with two copies of the nodal to orbital frame rotation matrix, $\mathcal{R}$, on the diagonal, and the $\mathcal{Z}$ terms are the canonical state vectors in the orbital frame [36]. The $\mathbf{p}$ and $\mathbf{q}$ terms are 3x1 vectors containing the momenta and coordinates from the local frame state vector, $\mathcal{L}$. The canonical equations of motion are then easily found by applying Hamilton's

Equations

$$\dot{q}_i = \frac{\partial \mathcal{K}_2}{\partial p_i}, \tag{18}$$

$$\dot{p}_i = -\frac{\partial \mathcal{K}_2}{\partial q_i}, \tag{19}$$

to the Hamiltonian function listed above at Equation 6 [10].

2.2 Floquet Theory

Once the equations of motion are known, they can be integrated for one period to obtain the desired periodic orbit. Linearized equations can then be generated by expanding about this periodic orbit. However, these equations are periodic in time, so the approach to solve them will be slightly more complicated than that of the Clohessy-Wiltshire solution. The solution used here is based on Floquet theory [32]. Floquet discovered that linearizing about a periodic orbit produces periodic coefficient linear systems. The time periodic linear differential equations, or variational equations, are

$$\delta\dot{\mathbf{x}} = A(t)\delta\mathbf{x}. \tag{20}$$

Floquet discovered that the solution of Equation 20 is the state transition matrix and can be defined as

$$\Phi(t,0) = F(t)e^{Jt}F^{-1}(0), \tag{21}$$

where F(t) is a periodic matrix, and J is a Jordon normal form matrix of the system frequencies. Usually J is diagonal, and its diagonal elements are termed Poincare exponents. This form is similar to the familiar form of the state transition matrix for constant coefficient systems. The only difference is that the F matrix is periodic in time.

To obtain $\Phi(t,0)$, the periodic matrix F(t) and J must be found. This is done by calculating the monodromy matrix, $\Phi(\tau,0)$. Equation 20 can be integrated for one period to find the monodromy matrix and the periodic orbit. Thus, the work of

finding the monodromy matrix is already accomplished from setting up the problem since the equations of variation were integrated once to obtain the reference periodic orbit. Remembering that F is periodic in time, Equation 21 at time τ becomes

$$\Phi(\tau, 0) = F(\tau)e^{J\tau}F^{-1}(0) \tag{22}$$

and upon some simple mathematical operations it can be seen that F(0) is just the eigenvector matrix of $\Phi(\tau, 0)$. Thus,

$$F^{-1}(0)\Phi(\tau, 0)F(0) = e^{J\tau}. \tag{23}$$

These eigenvalues of the monodromy matrix, called the characteristic multipliers, can be calculated at F(0) via

$$\lambda_i = e^{\omega_i}\tau, \tag{24}$$

where ω_i are the Poincare exponents found in J. Thus, the elements, and then J itself, can be found by rearranging Equation 24 to yield

$$w_i = \frac{1}{\tau}\ln\lambda_i. \tag{25}$$

By substituting the solution found at Equation 21 into the periodic differential equations of the system and capitalizing on the information seen above at Equation 25 about J to simplify things, the following result is obtained.

$$\dot{F} = A(t)F(t) - F(t)J \tag{26}$$

Now, after integrating the above equation for one period, the solution matrix from Equation 21 can be calculated. However, arriving at this integration is not as simple as it may appear since there are two integrals of the motion and thus two pairs of zero Poincare exponents. This makes finding the eigenvectors associated with the

zero Poincare exponents, and thus the periodic modal matrix F, a more difficult task. The method by which these are identified can be found in the relative motion paper by Wiesel [36]. Once they are found, the periodic modal matrix can be formed and integration of Equation 26 can be performed.

The end result is that the inertial position vector of a satellite can be expressed as

$$\mathcal{I}(t) = \mathcal{R}_Z^{(2)} \left\{ \mathcal{N}_0(t) + (\mathcal{R}^{(2)})^T F(t) e^{Jt} F^{-1}(t_0) \mathcal{Z}(t_0) \right\}, \tag{27}$$

while the position relative to the periodic orbit in the orbital frame is given by

$$\mathcal{Z}(t) = F(t) e^{Jt} F^{-1}(t_0) \mathcal{Z}(t_0). \tag{28}$$

In this work the solutions obtained via Floquet theory, namely the reference nodal state vector $\mathcal{N}_o(t)$ and the periodic modal matrix $F(t)$, are reduced to their Fourier series coefficients. This is necessary to obtain a numerically integrable form. The method by which they were reduced is based on harmonic analysis, as described in Brouwer and Clemence [2]. The method of harmonic analysis is used to find numerical coefficients of series when the eccentricity becomes increasingly more than that of a near circular orbit or when time periodic functions are involved. It states that if ω is any independent angular variable and P is the periodic function of ω, then P can be expressed as

$$\begin{aligned} P(\omega) &= \frac{1}{2}c_0 + c_1 \cos(\omega) + c_1 \cos(2\omega) + \ldots + \frac{1}{2}c_n \cos(n\omega) \\ &+ s_1 \sin(\omega) + s_2 \sin(2\omega) + \ldots + s_n sin(n\omega), \end{aligned} \tag{29}$$

where

$$\begin{aligned} c_k &= \frac{1}{n} \Sigma_{j=0}^{2n-1} P(j\alpha) \cos(kj\alpha), k = 0, 1, 2, \ldots, n, \\ s_k &= \frac{1}{n} \Sigma_{j=1}^{2n-1} P(j\alpha) \sin(kj\alpha), k = 1, 2, \ldots, n-1, \end{aligned} \tag{30}$$

and

$$\alpha = \frac{2\pi}{2n}. \tag{31}$$

It must be noted that since $sin(n\alpha) = 0$, there is no expression for s_n.

2.3 GPS Concepts

The Global Positioning System (GPS) is an all-weather, 24-hour, absolute positioning service maintained by the DoD. GPS is based on the principle of distance equals rate times time. Using this principle to time tag its transmissions, a user's positional fix can be obtained using a minimum of four satellites to determine the four unknowns of the positioning problem. The four unknowns are the three position states in the ECEF (earth-centered, earth-fixed) frame and the time ambiguity (the offset between the receiver time and the satellite time). Levels of accuracy for the GPS Standard Positioning Service (SPS) are advertised to be no less than 100 meters (2 drms, 95 percent) horizontally and 156 meters (95 percent) vertically. With the recent removal of Selective Availability (SA), the advertised accuracy levels have increased to about 25 meters (2 drms, 95 percent) in the horizontal plane and 43 meters (95 percent) in the vertical plane [13]. It must be emphasized that these are the specifications that are specified in the GPS Interface Control Document (ICD) 200, and that typical performance is much better. Through personal experience of this researcher, GPS accuracy without SA is on the order of a few meters in the horizontal plane and about twice as bad in the vertical direction. Although much improved, even these performance numbers may not be accurate enough to use in the applications discussed this far, especially space-based radar.

2.3.1 DGPS. A significant improvement to GPS SPS accuracy can be achieved via Differential GPS (DGPS) methods. These methods are code-based, carrier-based, or a combination of the two called carrier-smoothing. Code-based techniques produce the least improvement to SPS while carrier-based provides the

most. However, this performance increase comes at a cost in system complexity. Code-based techniques use pseudoranges (to be discussed momentarily), while the carrier-based techniques also use information from the phase of the carrier signal. Of course, this implies the carrier-smoothed code techniques lie somewhere in the middle. By using code-based techniques a user can acquire positions to within 1-2 meters of the truth. Carrier-smoothing methods improve that to about .1-.5 meters, and pure carrier methods yield positions with error on the order of only .01-.05 meters. While this latter accuracy is impressive, it must be noted that this type of method is not as robust as the others and is much more sensitive to vehicle dynamics. Too much motion may cause the receiver to lose lock and require 1-3 minutes to re-acquire the signal [20].

As mentioned previously, a minimum of four satellites are required to determine the three dimensional position of a user and the time offset of the receiver clock. To do this, each GPS receiver set calculates a pseudorange (PR or ρ) for each satellite, or the approximate range between the user and the satellite. The reason the range is approximate, and thus a pseudorange, is because the time offset is not known. A GPS pseudorange can be represented mathematically by the following

$$\rho = r + c(\delta t_u - \delta t_{sv} + \delta t_D), \tag{32}$$

where r is the true range, δt_u is the user clock error, δt_{sv} is the satellite clock error, and δt_D is the time error due to propagation delays [21]. With four pseudorange expressions from four different satellites, all four unknowns can be found. The time error due to propagation delays can further be broken down to its individual components due to SA, the troposphere (tropo), the ionosphere (iono), system noise (noise), multipath (mp), and receiver hardware noise (hw) via

$$\delta t_D = \delta t_{SA} + \delta t_{tropo} + \delta t_{iono} + \delta t_{noise} + \delta t_{mp} + \delta t_{hw}. \tag{33}$$

Through differencing methods, DGPS removes common errors from two or more receivers viewing the same satellites. There are two different types of differencing methods, single and double, and both methods simply difference measurements. Single differencing differences individual pseudoranges while double differencing differences two single difference measurements. Single differencing is easier to implement, however double differencing is used if very precise measurements are required. If the

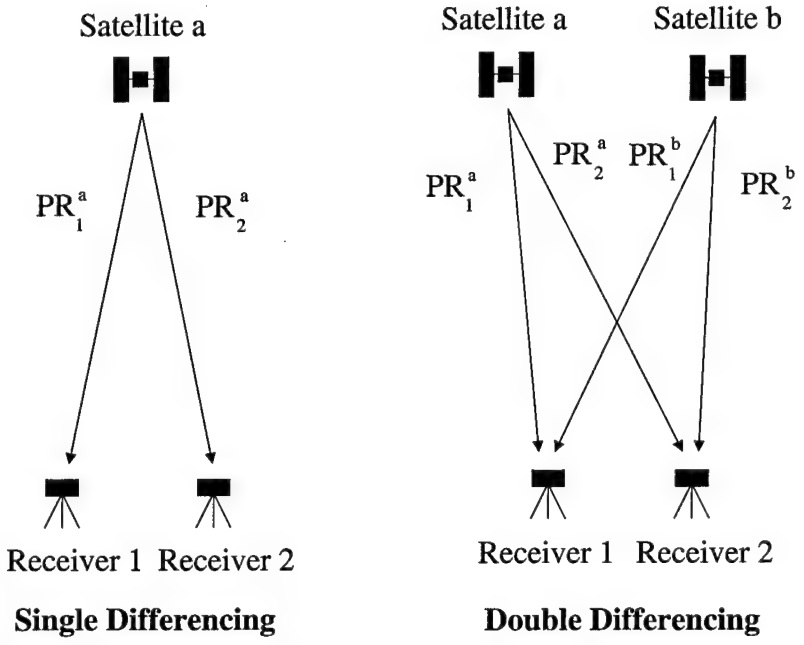

Figure 2 Differencing Methods

single difference measurement is defined as

$$\Delta\rho_{12}^{a} \equiv \rho_{1}^{a} - \rho_{2}^{a}, \tag{34}$$

then it follows that the double difference measurement is given by

$$\nabla\Delta\rho_{12}^{ab} \equiv \Delta\rho_{12}^{a} - \Delta\rho_{12}^{b} \tag{35}$$

$$
\begin{aligned}
&= \Delta r_{12}^{a} + \Delta c\delta t_{u_{12}} + \Delta c\delta t_{tropo_{12}}^{a} + \Delta c\delta t_{iono_{12}}^{a} + \Delta c\delta t_{noise_{12}}^{a} + \Delta c\delta t_{mp_{12}}^{a} \\
&- \Delta r_{12}^{b} - \Delta c\delta t_{u_{12}} - \Delta c\delta t_{tropo_{12}}^{b} - \Delta c\delta t_{iono_{12}}^{b} - \Delta c\delta t_{noise_{12}}^{b} - \Delta c\delta t_{mp_{12}}^{b} \\
\nabla\Delta\rho_{12}^{ab} &= \nabla\Delta r_{12}^{ab} + \nabla\Delta c\delta t_{tropo_{12}}^{ab} + \nabla\Delta c\delta t_{iono_{12}}^{ab} + \nabla\Delta c\delta t_{noise_{12}}^{ab} + \nabla\Delta c\delta t_{mp_{12}}^{ab}.
\end{aligned}
$$

Notice that the satellite clock error, receiver clock error and SA are not present in the double differenced measurement since these error sources were common between the two single differenced measurements. Double differencing techniques also have the benefit of reducing the tropospheric and ionosphereic errors. However, this is done at the expense of amplifying the multipath and noise by a factor of two [21].

Double differencing techniques are not unique to code-based methods (methods using pseudoranges). Carrier-based methods also use difference measurements. The rationale behind the development of carrier-phase differences is identical to that of the code-based difference measurements since carrier-phase measurements are subtracted to reduce errors from common sources (i.e. ionosphere, troposphere, etc.). While their derivation is similar in approach to the code-based differences, they are not derived here since the importance of why measurements are differenced has already been shown. The reader is directed to Kaplan [13] for their derivation.

2.3.2 CDGPS. To provide accuracy on the order of a few centimeters, CDGPS techniques incorporate the use of the signal carrier frequencies to provide phase information. These techniques are based on interferometric measurements of the satellite carrier frequencies. Measurements of the Doppler shift are taken, integrated to calculate the carrier advance at every epoch, and are then combined with receiver carrier-phase measurements to aid in the resolution of the carrier cycle ambiguity. Performance is on the order of 20 cm in dynamic applications [13]. This level of accuracy will increase as the random nature of the problem gives way to determinism. Thus, a system that is extremely deterministic may experience accuracy on the order of a few centimeters since the motion of the subject is limited

to more predictable paths. This statement confirms the need for a precise relative motion model for this work. Since the satellite formation dynamics used in this thesis are very *deterministic*, it could be expected that a generated CDGPS solution would have very little error induced by dynamics. Additionally, the formation's operating regime (i.e. space) is above the densest portion of the earth's atmosphere, so the potential for CDGPS-improved accuracy is further increased. Recall the expression for the code-based double difference measurement defined by Equation 36. Since the satellite formation will reside completely above the troposphere, the tropospheric delay term vanishes, further reducing error sources. It is estimated that the troposphere accounts for eight percent of position error for earth or near-earth applications [21]. Although this argument uses a code-based measurement for illustration rather than a carrier-based measurement, it is still correct since both types of difference measurements are generated with the goal to eliminate common error sources.

2.3.3 GPS in Space. Space-based GPS applications have been an avid area of research. Within the last few years, the focus from generic programs to relative spacecraft navigation has been pronounced. The Orion Project, supported by NASA Goddard Space Flight Center, is a concept demonstration that a cluster can use CDGPS and closed loop autonomous control to navigate the formation of spacecraft [14]. The main focus of any relative navigation problem is the search algorithm to resolve the integer carrier-phase ambiguity. Two such efforts have been accomplished by Cox and Binning [7, 1]. Mr. Cox's work [7] uses a combination of code and carrier-phase measurements in a Kalman filter. Some of the filter's states are dedicated to estimating the integer, carrier-phase ambiguities. In an effort to find the ambiguity set most probable and faster, the Least Squares Ambiguity Decorrelation Algorithm (LAMBDA) method is employed. The LAMBDA method was developed by Peter Teunissen in 1993 and searches for the smallest constant-probability-density ellipsoid centered at the Kalman ambiguity vector estimate and

uses the integer vector found on its surface. The general idea is that the method decorrelates highly correlated phase ambiguities by performing an ambiguity transformation that attempts to diagonalize the ambiguity states covariance matrix [20]. While total diagnalization is not possible, the resulting covariance matrix is highly de-correlated. This result increases the probability that the correct ambiguity set is found and the process is sped up. While there has been a push to use GPS for orbit determination, one paper in particular [4] warns the GPS community of the dangers of relying solely on GPS for determining a satellite's position/velocity vector. This work emphasizes that GPS data, even precise CDGPS data, must be used in concert with a filtering algorithm that includes enough determinism to compensate for the poor semi-major axis determination provided by GPS solutions.

2.4 Bayesian Estimation

A Bayes filter was chosen as the heart of the estimator for this work. A Kalman filter probably could have worked as well and thus would have echoed the approach of others in similar efforts, however a Bayes filter was deemed a much more natural choice due to the deterministic nature of this problem. Additionally, the Bayes algorithm is simpler to implement, considering a Kalman Filter may have a hard time with the extreme accuracy of the data and thus would require additional effort to correct for this shortcoming [33, 17].

The Bayes filter is a type of recursive filter, which in its simplest form reverts to a least squares form. The Bayes filter algorithm used in this thesis was adapted from Dr. Wiesel's class notes from MECH 731, Modern Methods of Orbit Determination [33]. However, there is nothing unusual about this formulation and it should echo any other estimation text on the subject. The following text will illustrate the overall flow of the Bayesian algorithm at the heart of this effort's estimator. At this point, only the shell of the filter needs to be presented. Specific details concerning the construction of the state vector, the observation relation, the linearization matrix

and the data to be used is covered in the Methodology section. The flow of the Bayesian algorithm is represented by Figure 3. First, the old estimate of the state

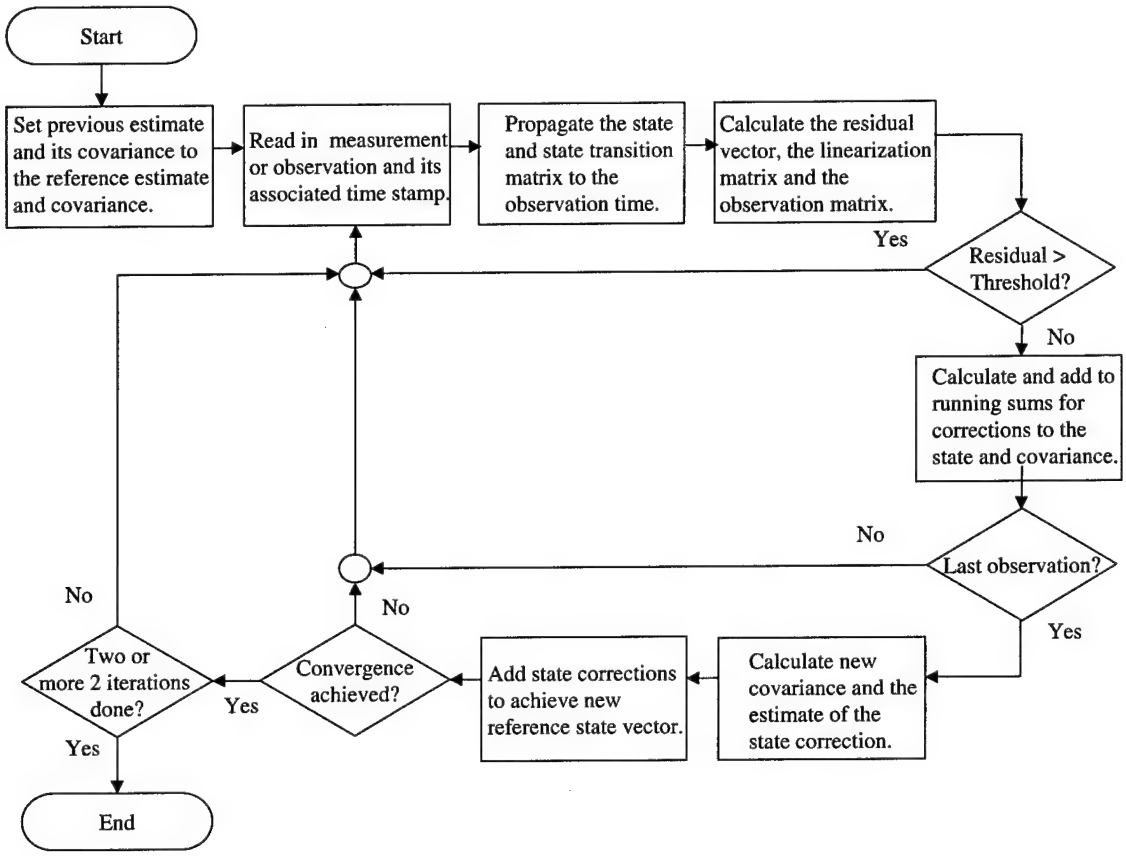

Figure 3 Bayesian Algorithm

and its covariance need to be advanced to the new epoch time. The previous estimate becomes the reference estimate by setting

$$\mathbf{x}_{ref} = \bar{\mathbf{x}}(-) \tag{36}$$

for each new observation time t_i. Next, the state and state transition matrix, Φ, are propagated to time t_i. The residual vector, r_i, the linearization matrix, H_i, and the

observation matrix, T_i, are then calculated by the expressions

$$\mathbf{r}_i = \mathbf{z}_i - \mathbf{G}(\mathbf{x}, t), \tag{37}$$

$$H_i \equiv \frac{\delta \mathbf{G}}{\delta \mathbf{x}}(\mathbf{x}_{ref}(t_i), t_i), \tag{38}$$

$$T_i \equiv H_i \Phi(t_i, t_0), \tag{39}$$

where $\mathbf{G}(\mathbf{x}, t)$ is the observation relation correlating the states to the measurement data. Then, the newly calculated terms are added to the sums below

$$\sum_i T_i^T Q_i^{-1} T_i, \tag{40}$$

$$\sum_i T_i^T Q_i^{-1} \mathbf{r}_i, \tag{41}$$

where Q is the observation covariance matrix associated with measurement noise.

At this point, any residuals greater than a predetermined threshold (expressed in multiples of standard deviations) are rejected. The previous steps are repeated until data is processed. Then the new covariance of the state correction is obtained by

$$P^{-1}(+) = P^{-1}(-) + \sum_{i=1}^{N} T_i^T Q_i^{-1} T_i. \tag{42}$$

Also, the estimate of the state correction is found via

$$\delta \bar{\mathbf{x}}(t_0) = P(+) \left(P^{-1}(-)(\bar{\mathbf{x}} - \mathbf{x}_{ref}) + \sum_{i=1}^{N} T_i^T Q_i^{-1} \mathbf{r}_i \right). \tag{43}$$

The last calculation then corrects the reference solution according to

$$\mathbf{x}_{ref+1}(t_0) = \mathbf{x}_{ref}(t_0) + \delta \bar{\mathbf{x}}(t_0). \tag{44}$$

Finally, convergence is checked. If convergence is not achieved, then the whole process starting after the first step, is repeated. Conversely, if there is convergence and

at least two iterations have been performed, then $\mathbf{x}_{ref+1}$ becomes the new estimate with covariance P(+).

III. Methodology

3.1 Problem Definition

The main objective of this thesis is to provide centimeter level accuracy in the estimation of relative satellite positions. Further, it is desired to separate the absolute positioning problem from the relative. This will be attempted by using only relative positional information. Estimates of the absolute measurements on the order of tens of meters or worse will suffice if the estimates of the relative positions can be obtained on the order centimeters (the magnitude of the noise). Before any attempt to achieve these objectives, preliminary work is required. This work includes selecting a formation design, to include the number of satellites, their geometry and the reference periodic orbit. Also, the basic Bayesian filtering code and the truth model for measurement generation must be developed and tested. This section will explain these details as well as describe the state vector and the specific filters used in this thesis.

3.1.1 Satellite Formation Selection. This work will use a cluster composed of three satellites. Since this is an arbitrary number, it is a matter of simple expansion to include more than three. In fact, the filter code is written to incorporate any number of satellites. The only real limit on the cluster size is processing power since the covariance to be inverted grows by six rows and columns with every additional satellite. Thus, a ten satellite cluster would require an inversion of a 60 by 60 covariance matrix! This may be readily handled by any PC on the market, however processing power in space is a commodity that should not be unnecessarily wasted.

For the purposes of this thesis, the satellites will be in no specific formation. This is for ease of calculation and not likely to be the most optimal for mission design. This work is focused on estimating the relative positions of the elements of a formation with respect to each other and not the optimal design for the cluster

geometry. Thus, the only true design strategy implemented was to choose the cluster element positions such that no secular modes were excited. This was done to suppress secular terms that would encourage the formation to drift apart and was accomplished by setting the fourth and sixth Floquet modal variable (modal variables are discussed further in Section 3.1.2) to zero. The spacecraft are spaced about 200 meters from the periodic orbit vector, but in no definitive pattern (see Table 1 for approximate positions of each spacecraft in the CW frame). The periodic orbit vector has its origin at the center of the earth and its tip at the average position of the formation elements in the CW frame. The reference periodic orbit modeled is nearly circular and it has an altitude of 637 kilometers with inclination of one radian. Although several efforts have focused on formations with inclinations of 90 degrees to eliminate the nodal regression [23, 28], this simplification was avoided since most missions will not be polar. Additionally, the perturbations experienced by non-polar formations need to be investigated.

Table 1 Approximate Spacecraft Positions in CW Frame

	Radial (meters)	In-Track (meters)	Cross-Track (meters)
Satellite 1	-125	39	-147
Satellite 2	125	39	278
Satellite 3	-147	-48	-48

3.1.2 State Definition. The goal of this effort is to estimate the true element positions of a satellite formation to within a few centimeters. This implies that the states to be used are physical in nature and in a frame of choice (i.e. inertial, ECEF, orbital, CW, etc.). However, the problem reveals more about itself if states are chosen within phase space, or more specifically using Floquet modal variables. While abstract, these phase states tend to offer more information about the dynamical system than physical states.

The Floquet modal variables are essentially transformed orbital frame physical states. Thus, it follows from Section 2.2 that the Floquet modal variables have six

eigenvectors, one for each of the satellite's states. These eigenvectors form three modes. Three is expected rather than six (one for each eigenvector) since the real and imaginary parts of the complex modes in the formation of the modal matrix F in Section 2.2 are isolated according to method of Calico and Wiesel [35]. The three modes are the eccentricity mode, the energy mode, and the angular momentum mode. The first mode consists of the perturbations of the eccentricity of the periodic orbit and the advance of perigee. The second mode consists of the in-track velocity and period change of the periodic orbit. The third mode consists of the change in the node of the satellite relative to the periodic orbit and the change in the inclination. Thus, each satellite's state is proportional to the following

$$\mathbf{X}_{modal} = \begin{pmatrix} m_{1,1} \\ m_{1,2} \\ m_{2,1} \\ m_{2,2} \\ m_{3,1} \\ m_{3,2} \end{pmatrix} \propto \begin{pmatrix} \Delta e \\ \Delta\omega \\ \Delta v_{in-track} \\ \Delta T \\ \Delta\Omega \\ \Delta i \end{pmatrix}, \tag{45}$$

where m_{ij} represents the i_{th} complex mode with its two j parts, Δe is the perturbation of the eccentricity, $\Delta\omega$ is the advance of the argument of perigee, Δv is the in-track displacement, ΔT is the change in period, $\Delta\Omega$ is the change in the node, and Δi is the change in inclination. Wiesel shows that to prevent the formation from dispersing, the fourth and sixth state need to be kept to zero and the secular terms of the third and fifth suppressed. As long as the cluster states (i.e. geometry of the cluster) are chosen to limit the introduction of secular terms into the system, specifically in-track dispersion, the rest of the terms will have periodic fluctuations and the cluster will remain intact [35].

While the insight gained by using Floquet modal variables is extremely important, the end result is to know physical positions. Thus, once convergence in the

filter is achieved the physical states must be obtained through the transformation that was alluded to earlier. The relation between the physical state and the modal state is

$$\mathbf{X}_{physical}(t) = \begin{pmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = F(t)e^{Jt}F(0)^{-1}\mathbf{X}_{physical}(t_0) = F(t)e^{J(t)}\mathbf{X}_{modal}(t_0). \quad (46)$$

At this point, it is important to point out a subtle peculiarity found within each filter's relative dynamics routines investigated by this thesis. Since the filters use modal states instead of physical states within their relative dynamics models, they do not use the exact form of the State Transition matrix detailed in Section 2.2. Instead, each one uses a hybrid form shown in the latter part of Equation 46. The effective State Transition matrix of each filter is explicitly given by

$$\Phi_{filter} = \frac{\partial x(t)_{physical}}{\partial x(t_o)_{modal}} = F(t)e^{J(t)}. \quad (47)$$

This is only slightly different than the standard definition given in Section 2.2 since that section defines it as

$$\Phi_{standard} = \frac{\partial x(t)_{modal}}{\partial x(t_o)_{modal}} = F(t)e^{J(t)}F(0)^{-1}. \quad (48)$$

Thus, in this work's filters, the effective form of the State Transition matrix serves two purposes. In addition to the traditional role of propagating the state vector, it also represents the transformation matrix between the Floquet modal variables and the physical states.

3.2 Initial Bayes Filter Model

An initial Bayes Filter was built to demonstrate the functionality of the algorithm detailed in Section 2.4. A simple test case of a linearly increasing, two-state system with constant acceleration was chosen as the test system. In addition to the equations in Section 2.4 required to execute the Bayesian algorithm, a dynamics routine and observation relation processor were created to reflect this simple system. The sample runs only consisted of a handful of data points, and tests included perfect data and noisy data. The perfect data yielded zero residuals and convergence while the noisy data caused the estimator to produce appropriately sized residuals and state corrections.

3.3 Truth Model

A truth model is used to output not only the true state but also corrupted data for the filter. The truth model interacts with the filter according to Figure 4. The truth model contains the equations of motion and variation for the two body motion in the reference periodic orbit discussed previously. The equations of motion are developed in Section 2.1.2, and the equations of variation are found by linearizing about the equations of motion. The initial states of the system, output data noise values, and other initialization flags and variables are stored in a file represented as the Sample Run Parameter File in Figure 4. This file is used to initialize the truth model and allow it to generate data consistent with the goals of a particular data run. As previously mentioned, the equations of motion are integrated for one orbit to obtain the periodic orbit. This solution is reduced to its Fourier series coefficients and is stored in a file that is represented in Figure 4 as the Reference Orbit Data File. This file also contains the Fourier series representation of the Floquet solution as well. This data is used by both the truth model and the Bayes filter. While the truth model and Bayes filter both use this data for predictions of the system dynamics, the truth model uses numerical integration as opposed to the filter's linear algebra techniques.

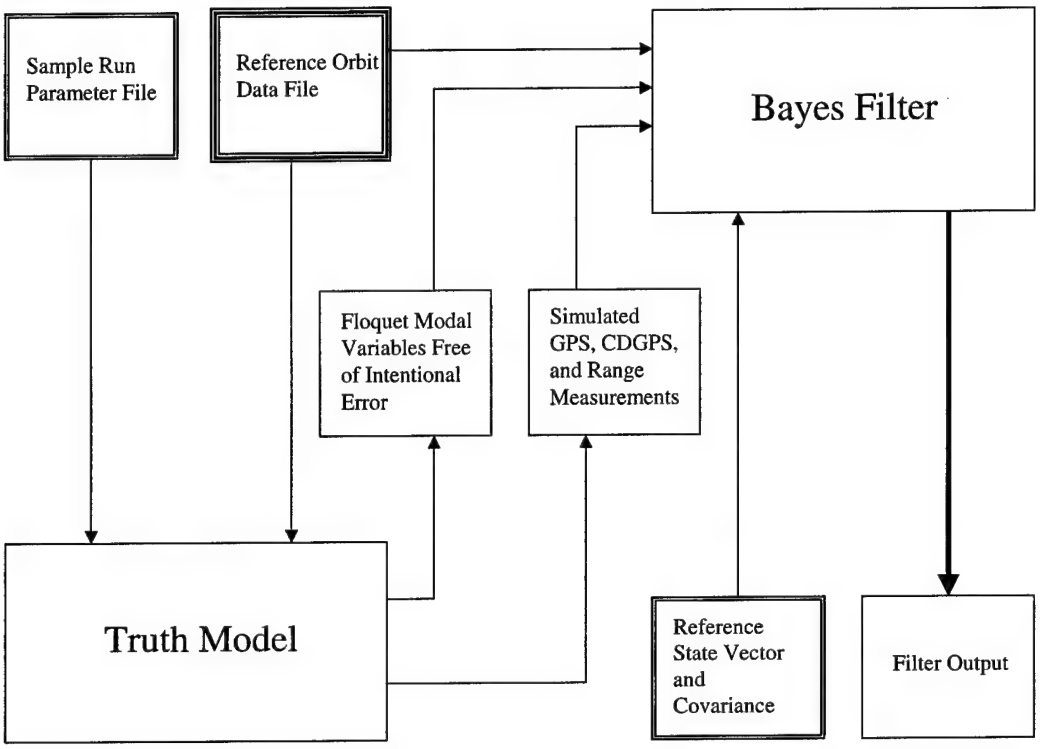

Figure 4 Truth Model and Bayes Filter Interaction

The truth model uses a Hamming predictor-corrector algorithm to find the system states at time intervals specified in the input file. The truth model can handle a user specified level of zonal harmonics of the earth in its calculations. Although this work used zonal terms up to the 14th order, the order can be increased, or decreased, depending on the desired modeling needs of the simulation. Finally, the truth model also incorporates the earth's tesseral and sectoral harmonics and air drag upon user request.

The truth model manipulates the states found via integration to form the two types of data output to the filter, free modal variables and measurements. The free modal variables are the Floquet modal variables without any intentional noise added.

The measurements used in this work include GPS, CDGPS, and range data generated from internal clock pulses by each satellite. These three types of measurements, $\mathbf{Z}_i$, are related to the physical states according to the following expressions. For the DGPS measurements the relation is

$$\mathbf{Z}_{CDGPS} = \begin{pmatrix} x_1 - x_i \\ y_1 - y_i \\ z_1 - z_i \end{pmatrix}, \tag{49}$$

where $i = 2, 3, \ldots, n$ for n satellites. The range measurements are represented by

$$\mathbf{Z}_{range} = |\vec{r}_i - \vec{r}_j| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \tag{50}$$

where $i = 1, 2, 3, \ldots, n$, $j = 2, 3, \ldots, n$, $i \neq j$, and n is the number of satellites. The last measurement output from the truth model is that of a GPS measurement for satellite one and it can be represented as

$$\mathbf{Z}_{1GPS} = \vec{R}_{absolute} - \vec{R}_{po} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix}, \tag{51}$$

where $\vec{R}_{absolute}$ is the absolute position vector to the satellite in the orbital frame and $\vec{R}_{po}$ is the position vector of the periodic orbit (see Figure 5). This GPS measurement introduced an absolute flavor to the filter's measurements. The reasons for doing this will be described momentarily in Section 3.4.1. As mentioned earlier, the truth model applies Gaussian noise to the output measurements upon user request, resulting in

$$\mathbf{Z}_{noisy} = \mathbf{Z}_{perfect} + \sigma_{noise}. \tag{52}$$

Normally, the input to the filter is in the form of the above equation. However, during testing the noise was removed for analysis and debugging. σ_{noise} represents

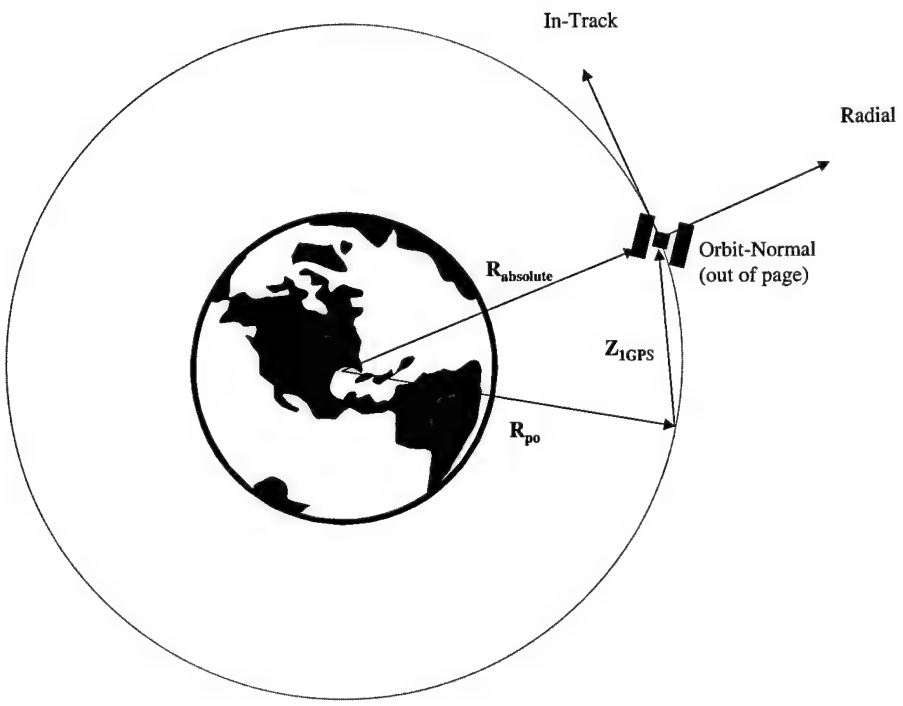

Figure 5 GPS Measurement

the Gaussian noise associated with the instrument covariance, Q, used in the Bayes algorithm. The noise represents an estimate of the errors found in each of the position or range determining systems. The noise at each time step is generated by a Gaussian pseudo random number generator. The magnitude of the error is within a user specified, integer multiple of an estimated standard deviation bound for each given measurement source. The user input corresponds to the instrument noise covariance matrices embedded in the filter code.

The instrument noise covariance matrices are populated with variances corresponding to each data type. The simulated CDGPS measurements', GPS measurements', and range measurements' noise are estimated to have a 1-sigma error bound ($\sigma_{data-type}$) of 3 centimeters, 3 meters, and 2 centimeters in each axis of the CW frame, respectively. The assumption of 3 centimeter error in each axis for the CDGPS measurements is consistent with, and in some cases more conservative

than, several GPS papers showing the potential accuracies of such measurements [6, 1, 19, 7]. The resulting instrument covariance matrix, Q, for each measurement type can be represented as

$$Q = \begin{bmatrix} \sigma^2_{data-type} & 0 & \dots & 0 \\ 0 & \ddots & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \dots & 0 & \sigma^2_{data-type} \end{bmatrix}. \tag{53}$$

Since the instrument covariance was created as a diagonal matrix, the measurements are modeled as having uncorrelated errors. While this may not be completely accurate, it is a reasonable approximation since any correlations of error within the measurements should not be very large or persistent.

3.4 Filters

Once the initial filter algorithm described in Section 3.2 was validated, the simple linear system was replaced with the system of interest to this thesis, that of a satellite formation. The perturbations modeled within this system are the earth's zonal harmonics. Although the truth model simulates tesseral, sectoral, and air drag perturbations, they were not included and are recommended for future work. In all, three different filters were investigated. Two versions were planned in advance and the third was included as a consequence of findings during implementation. Each filter differs from the others in the way its dynamics routine and observation relation processor are constructed.

In each filter, the dynamics routine is based on the theory presented in Section 2.1.2 and transformation (from the Floquet modal variables to physical states) described in Section 3.1.2. Due to this previous description, it is not covered in much detail here. While all the filters have the same dynamics model at their heart, not all of the filters propagate the same states. However, each are alike in that the

dynamics routine takes in the modal reference state vector and current time step as input variables, and returns the physical states and state transition matrix at the current time step.

While the dynamics routine is specific to the relative satellite navigation problem in its design and thus required massive overhaul from the simple linear system, the observation relation routine is created using the same approach used to formulate $\mathbf{G}(\mathbf{x}, t)$ and $\mathbf{H}$ before (for the simple linear system). $\mathbf{G}(\mathbf{x}, t)$ is still the function that describes the measurements in terms of the states and $\mathbf{H}$ is found by taking partial derivatives of $\mathbf{G}(\mathbf{x}, t)$ with respect to each element of the state. In other words,

$$H_i \equiv \frac{\delta \mathbf{G}}{\delta \mathbf{x}}(\mathbf{x}_{ref}(t_i), t_i). \tag{54}$$

The observation relation processor in each filter is based on the algorithms discussed in the Bayesian Estimation section. This processor takes the physical states output by the dynamics routine and forms estimates of the measurements at the given time step in the manner described in Section 3.3. The following subsections explicitly show $\mathbf{G}(\mathbf{x}, t)$ and $\mathbf{H}$ for three satellites for each filter design and points out any other significant differences.

3.4.1 Two Relative Data Filters - CDGPS and Range Data. The initial measurements used in this thesis were selected to be purely relative due to lessons learned in previous work performed by Captain Davis. He found that absolute measurements wasted precious significant digits since the measurements started at the center of the earth. Thus, the covariance became excessively large and would not invert. If the cluster element positions are measured from a closer reference (in this work the reference periodic orbit), more significant digits are available to allow precision on the order of centimeters or less without inversion problems. Hence, CDGPS measurements and range measurements between the cluster elements were the only measurements initially used. While it is true that CDGPS measurements

are absolute in nature, they were fed into the filter as a differenced measurement (recall Equation 49) and the measurements themselves are absolute only in a relative sense to the other satellite formation elements. For the range data the relations are

$$\mathbf{G}_{Range} = \begin{Bmatrix} \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \\ \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2} \\ \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2} \end{Bmatrix} \tag{55}$$

and

$$\mathbf{H}^T_{Range} = \begin{Bmatrix} \frac{x_1 - x_2}{\rho_{12}} & \frac{x_1 - x_3}{\rho_{13}} & 0 \\ \frac{y_1 - y_2}{\rho_{12}} & \frac{y_1 - y_3}{\rho_{13}} & 0 \\ \frac{z_1 - z_2}{\rho_{12}} & \frac{z_1 - z_3}{\rho_{13}} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{-(x_1 - x_2)}{\rho_{12}} & 0 & \frac{x_2 - x_3}{\rho_{23}} \\ \frac{-(y_1 - y_2)}{\rho_{12}} & 0 & \frac{y_2 - y_3}{\rho_{23}} \\ \frac{-(z_1 - z_2)}{\rho_{12}} & 0 & \frac{z_2 - z_3}{\rho_{23}} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{-(x_1 - x_3)}{\rho_{13}} & \frac{-(x_2 - x_3)}{\rho_{23}} \\ 0 & \frac{-(y_1 - y_3)}{\rho_{13}} & \frac{-(y_2 - y_3)}{\rho_{23}} \\ 0 & \frac{-(z_1 - z_3)}{\rho_{13}} & \frac{-(z_2 - z_3)}{\rho_{23}} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{Bmatrix}, \tag{56}$$

where ρ_{ij} is the range between the satellite pair. For the relative DGPS data the relations are

$$\mathbf{G}_{DGPS} = \left\{ \begin{array}{c} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \\ x_3 - x_1 \\ y_3 - y_1 \\ z_3 - z_1 \end{array} \right\} \tag{57}$$

and

$$\mathbf{H}_{DGPS} = \left\{ \begin{array}{cccccccccccccccccc} -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right\}. \tag{58}$$

Initial testing using only two satellites showed that the covariance would not invert using only CDGPS or range data since these measurements led to half the states being unobservable. The observability problem is not an issue for the range data when more than two satellites are modeled, but is always present for the CDGPS data no matter how many satellites are used. Since the modal variables for two satellites propagate as

$$\begin{aligned} \mathbf{X_1}(t)_{modal} &= e^{Jt}\mathbf{X_1}(t_0)_{modal}, \\ \mathbf{X_2}(t)_{modal} &= e^{Jt}\mathbf{X_2}(t_0)_{modal}, \end{aligned} \tag{59}$$

and the orbital position and velocities are found by

$$\mathbf{X(t)}_{physical} = F(t)\mathbf{X(t)}_{modal}, \tag{60}$$

the difference in the orbital frame position and velocity vector propagates as

$$\begin{aligned}\Delta\mathbf{X}(t)_{physical} &= \mathbf{X_2}(t)_{physical} - \mathbf{X_1}(t)_{physical} \\ &= F(t)\left(\mathbf{X_2}(t)_{modal} - \mathbf{X_1}(t)_{modal}\right) \\ &= F(t)e^{Jt}\left(\mathbf{X_2}(t_0)_{modal} - \mathbf{X_1}(t_0)_{modal}\right). \end{aligned} \tag{61}$$

These equations show that both sets of $\mathbf{X_i}(t_0)_{modal}$ cannot be determined, only their difference. This means one would expect six zero eigenvalues in the covariance matrix of the states, or half the states being unobservable, in a twelve state system. This argument was verified using singular value decomposition analysis. Only six non-zero singular values were found for the twelve-state system.

Absolute measurements were introduced to resolve the observability problem for both relative data filters when only two satellites were modeled and for the CDGPS filter when any number were modeled. The absolute measurement chosen was the absolute position of the satellite designated as the leader, satellite one. In a real estimation system, each element of the cluster would perform as if it were the "leader" satellite. This absolute measurement was fed into the estimator in the form of a GPS measurement, however it was massaged into a pseudo-relative form prior to being used to avoid the inversion problems encounterd before by using absolute measurements. The measurement can be represented as

$$\mathbf{Z_{1}}_{GPS} = \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \vec{R}_{absolute} - \vec{R}_{po}, \tag{62}$$

where $\vec{R}_{absolute}$ is the absolute position vector to the satellite in the orbital frame and $\vec{R}_{po}$ us the position vector of the periodic orbit (see Figure 5). For the absolute

GPS data the measurement relations are

$$\mathbf{G}_{GPS} = \left\{ \begin{array}{c} x_1 \\ y_1 \\ z_1 \end{array} \right\} \tag{63}$$

and

$$\mathbf{H}_{GPS} = \left\{ \begin{array}{cccccccccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right\}. \tag{64}$$

3.4.2 Relative State Filter - CDGPS Data. As a result of the initial failure using purely relative CDGPS data, it was shown earlier that the differences in the physical states could be propagated via

$$\Delta\mathbf{X}(t)_{physical} = F(t)e^{Jt}\left(\mathbf{X_2}(t_0)_{modal} - \mathbf{X_1}(t_0)_{modal}\right). \tag{65}$$

Thus, a new filter that used CDGPS measurements was constructed with the above expression as its dynamics routine. CDGPS measurements were chosen arbitraily; range measurements will work just as well. This new filter, termed the relative state filter, has a state vector that is represented as the differences in the modal states between two satellites rather than the modal states themselves. For the two satellite system used, this change reduced the number of states from twelve to six. The state

vector for this differential filter is given by

$$\Delta\mathbf{X}_{modal} = \begin{pmatrix} X_{2_1} - X_{1_1} \\ X_{2_2} - X_{1_2} \\ X_{2_3} - X_{1_3} \\ X_{2_4} - X_{1_4} \\ X_{2_5} - X_{1_5} \\ X_{2_6} - X_{1_6} \end{pmatrix}_{modal} . \tag{66}$$

Since the dynamics relation is shown by Equation 65, the only thing left to define is the observation relation correlating the states to the measurements. After propagating the physical states at each time step, it can be seen that the first three terms of the relative *physical* state vector is already in the form of the observation relation for the CDGPS data defined previously.

$$\Delta\mathbf{X}_{physical(1,2,3)} = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{pmatrix}_{physical} = \mathbf{G}_{CDGPS} = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{pmatrix} \tag{67}$$

Therefore $\mathbf{H}$ for a two satellite system using CDGPS measurements is just

$$\mathbf{H}_{GPS} = \begin{Bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{Bmatrix} . \tag{68}$$

3.5 Data Runs

Data runs are accomplished with each of the three filters: the relative data filter using range data, the relative data filter using CDGPS and GPS data, and the relative state filter using CDGPS data. Each run uses 24-hours worth of data with the data generated at 100 evenly spaced intervals (every 14.4 minutes). Each

data run is accomplished twice. First the filter is fed data with no noise, and then the filter ran again with data having noise appropriate to the measurement type (see Section 3.3) to simulate a more realistic response. The filters are summarized pictorially in Figure 6.

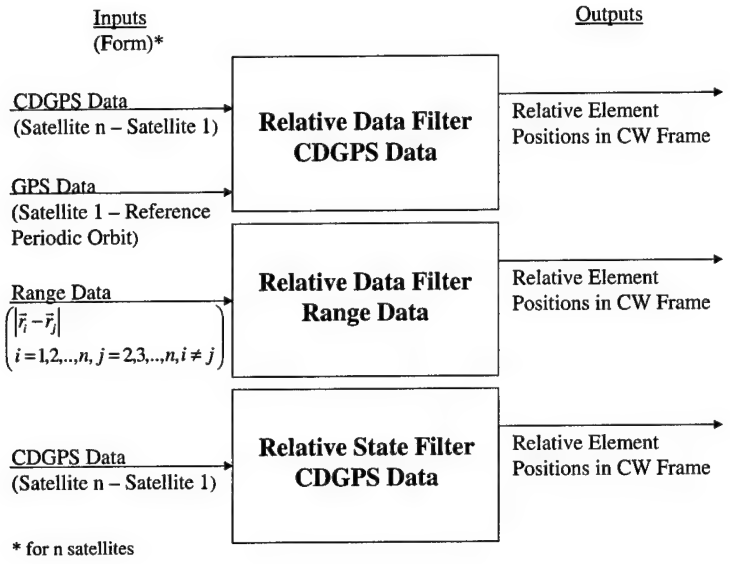

Figure 6 Filter Diagrams Showing Inputs and Outputs

IV. Results and Analysis

Now that the filter has been completed and its proper functioning verified, data runs can be accomplished and the results analyzed. Examination of the residuals, or the difference between the true observations and the filter's predicted ones, can lead to significant insight into the performance of the filter. Ideally, residuals should be random and small, thus any pattern, bias, or anomalous residual sizes should be investigated. The following sections first investigate filter residuals using noise free measurements. If all is working correctly, the residuals will show the amount of error in the system. This should reflect the noise floor set by the noise covariance matrices in the filter code for each respective measurement type. Once the behavior of the system is characterized without noise, plots of the residuals with noise added are analyzed. All plots shown in this chapter are with respect to the CW frame (radial (x), in-track (y), and cross-track(z)). Overall, the filters performed well but worse than expected under initial designed conditions. Since residuals are not expected when using error-free measurements (or at least not large ones), an error source, or sources, was determined to be at work in the production of the residuals shown in the error-free measurement plots. The filter algorithm was verified, thus it was not identified as an error source. Instead, the error was attributed to differences in the relative motion solution between the filter and the truth model as well as effects from linearization. The error sources will be discussed in further detail, where appropriate, in the subsequent discussions.

4.1 Relative Data Filter - CDGPS Filter

Figures 7 and 8 show the first pass residuals for the CDGPS filter's CDGPS measurements and GPS measurements, respectively. In both figures, the in-track measurements appear to walk away from zero rather quickly and secularly. The residuals also show periodic fluctuations in their values. Both of these trends may

be attributed to a reference periodic orbit mismatch (to be discussed in more detail later) since the filter and truth model see a different reality. These trends are evident in this filter's residuals as well as the other two studied in this work. Figure 9 is a tighter look at the data in Figure 8. It shows the behavior of the radial and cross-track residuals for the GPS measurements. While the in-track measurements diverge quickly (as seen in Figure 8), these two measurements oscillate about zero and gain error at a much slower pace.

First pass residuals do not mean too much since the filter has not yet converged. However, the trends of the first pass residuals will affect how the filter will converge and thus will impact the last pass residuals. Figure 10 and Figure 11 show the last pass residuals for the CDGPS filter's CDGPS measurements and GPS measurements, respectively. Figure 12 is a close up of Figure 11. After 24-hours worth of measurements, the residuals appear to still be diverging. However, the rate is much slower than that seen in the first pass residuals. The filter has bounded the error to be less than about 30 cm. The satellite pair comprised of 1 and 3 seem to perform marginally better than that of 1 and 2. This gain in performance can be attributed to the geometry of the cluster. Satellites 1 and 3 are closer together than satellites 1 and 2. The last pass GPS residuals no longer diverge, however they oscillate about zero error with max magnitudes near 500 meters. This poor performance in the absolute positioning is due to the reference periodic orbit mismatch stated earlier. The truth model integrates the formation's equations of motion, calculates the physical states at every time interval, and subtracts the assumed reference periodic orbit to obtain absolute GPS measurements with respect to the periodic orbit. The filter propagates the states via the state transition matrix, which is based on the same assumed periodic orbit. The states obtained are in the form of absolute GPS measurements with the reference orbit subtracted, thus they are identical in form to the truth model GPS measurements and no further manipulation is needed. Both the filter and the truth model use the same Fourier series coefficient representation of the

reference periodic orbit in the aforementioned calculations, however the difference in the implementation appears to be causing error in the calculation of the states. Specifically, the brute force integration of the equations of motion is showing that the assumed reference periodic orbit used in each filter is not the *true* periodic orbit of the system. Consequently, large oscillating residuals are seen in the absolute data (Figure11 and Figure 12) and small oscillations are seen in all of the relative data plots. Figure 13 and 14 shows the filter's performance when noise was inserted into the measurements. The performance was almost identical to that of the error-free measurements.

Overall, the filter converged to a solution with residuals much greater than were expected. In fact, even with error free measurements the residuals for each measurement type are larger than their corresponding measurement covariances programmed into each filter. Residuals were expected to be much smaller and only due to numerical or minor filter/truth model differences. Even with error introduced by error sources other than predetermined measurement noise, the residuals should have been no larger than the order of the noise of each measurement type expected by the filter - 3 centimeters in each axis for the CDGPS measurements (1-sigma) and 3 meters in each axis for the GPS data (1-sigma). The fact that large residuals were encountered indicates that the filter's dynamics model has a flaw. The large residuals in the absolute measurements are not related to this flaw since they have been shown to be caused by the reference periodic orbit mismatch. The flaw in the filter's dynamics model was initially identified as the lack of second order dynamics in the solution. The reference periodic mismatch may be affecting the relative measurement residuals to some extent, but the largest contributor to the unexpected size of these residuals was believed to be caused by second order effects.

To prove this theory, the cluster spacing was reduced by an order of magnitude (see Table 2 for approximate positions of each spacecraft after reduction of orbital spacing). Larger residuals can be expected if the second order terms (and higher)

neglected in the linearization of the filter's dynamics are not negligible. It was expected that if this was the case, reducing the spacing between the spacecraft by an order of magnitude would decrease the residuals by two orders of magnitude or at least to the noise floor set by the filter for each type of measurement. Figures 15 through 17 show the results of the first pass. Figures 18 through 20 show the results of the last pass.

Table 2 Approximate Spacecraft Positions In Reduced Formation

	Radial (meters)	In-Track (meters)	Cross-Track (meters)
Satellite 1	-12.5	3.9	-14.7
Satellite 2	12.5	3.9	27.8
Satellite 3	-14.7	-4.8	-4.8

The reduction in orbital spacing has created residuals with magnitudes that were expected for the CDGPS measurements, about 3 cm in each axis (1-sigma). However, it is seen that the GPS measurement residuals did not change. This is expected since the predominant error source in the relative measurement residuals is the lack of second order dynamics while the absolute estimates are suffering from the reference periodic orbit mismatch mentioned earlier. Figures 21 and 22 shows the filter's performance when noise was added to the measurements. The performance of absolute positioning was nearly identical to that of the error free data and the relative measurement residuals seem to be still largely affected by error other than from the measurements. This means that modeling error is the dominating source of error in the filter's position estimates and *not* sensor noise.

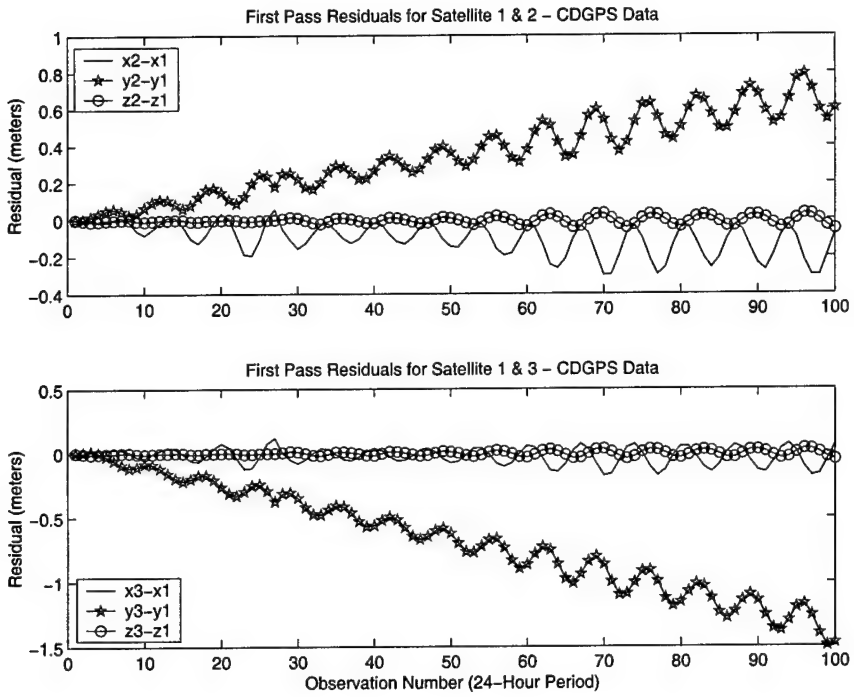

Figure 7 CDGPS Filter's First Pass Residuals of CDGPS Data (Noise Free Data and Original Orbital Spacing)

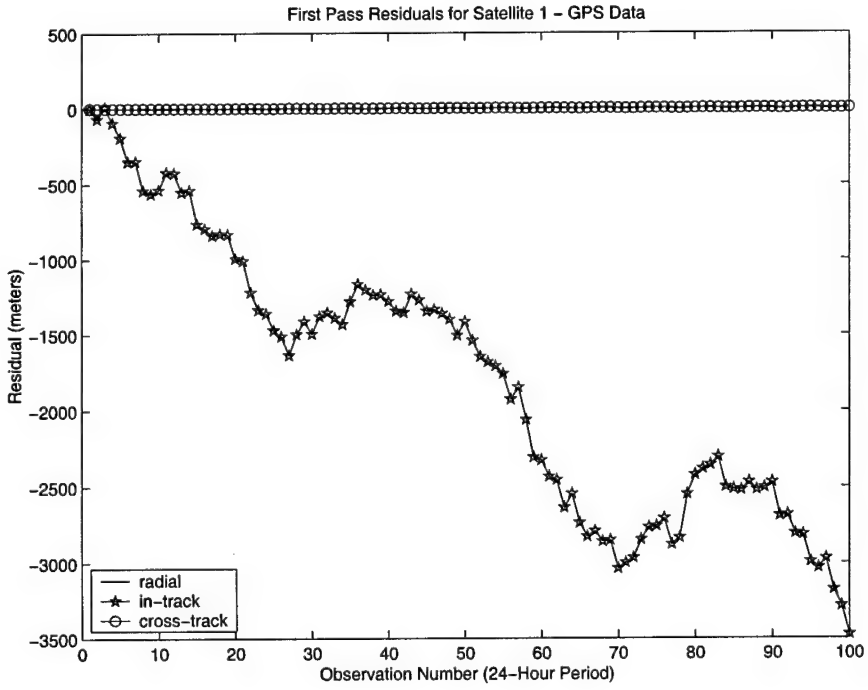

Figure 8 CDGPS Filter's First Pass Residuals of GPS Data (Noise Free Data and Original Orbital Spacing)

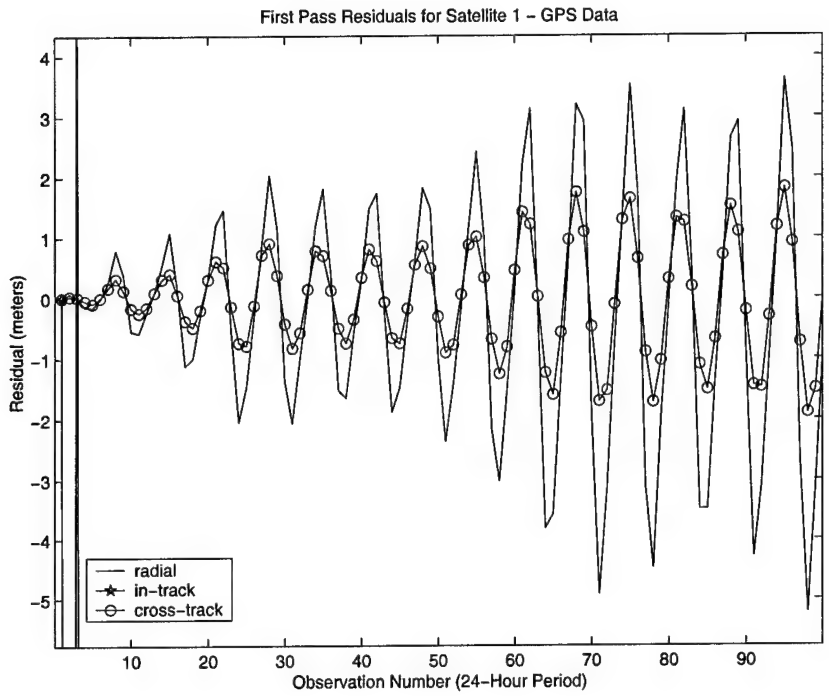

Figure 9 Zoom In of CDGPS Filter's First Pass GPS Residuals (Noise Free Data and Original Orbital Spacing)

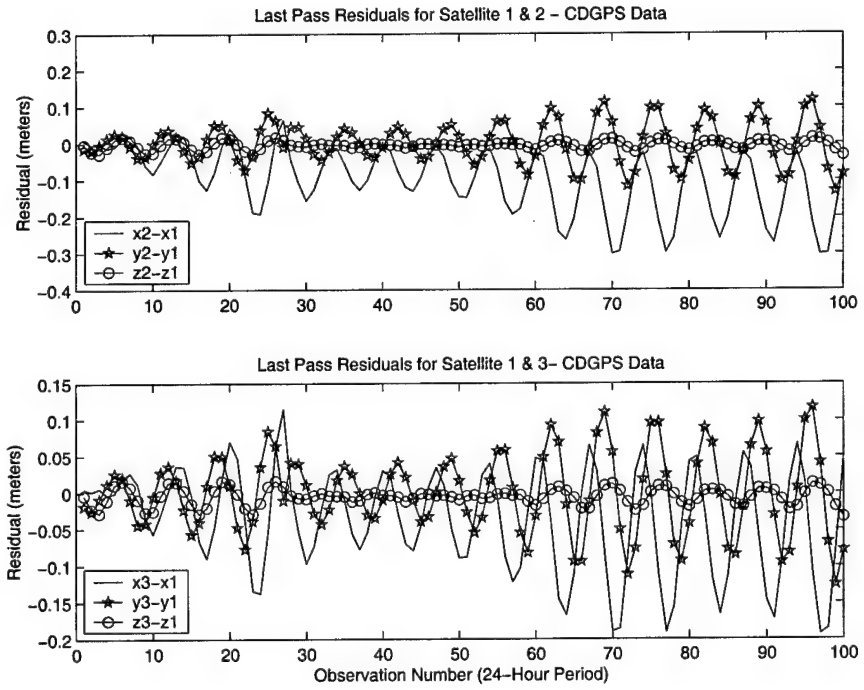

Figure 10 CDGPS Filter's Last Pass Residuals of CDGPS Data (Noise Free Data and Normal Orbital Spacing)

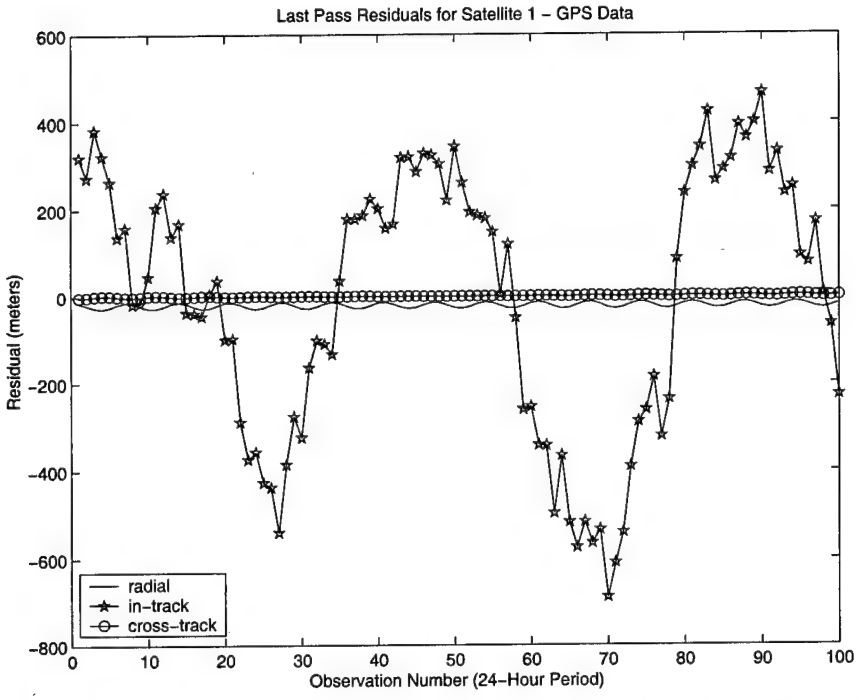

Figure 11 CDGPS Filter's Last Pass Residuals of GPS Data (Noise Free Data and Normal Orbital Spacing)

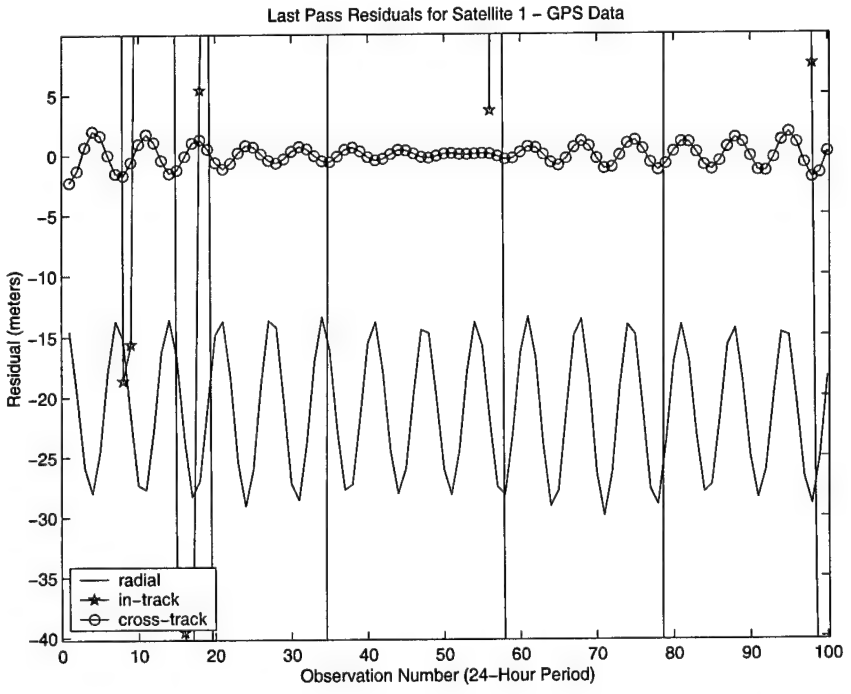

Figure 12 Zoom In of CDGPS Filter's Last Pass GPS Residuals (Noise Free Data and Normal Orbital Spacing)

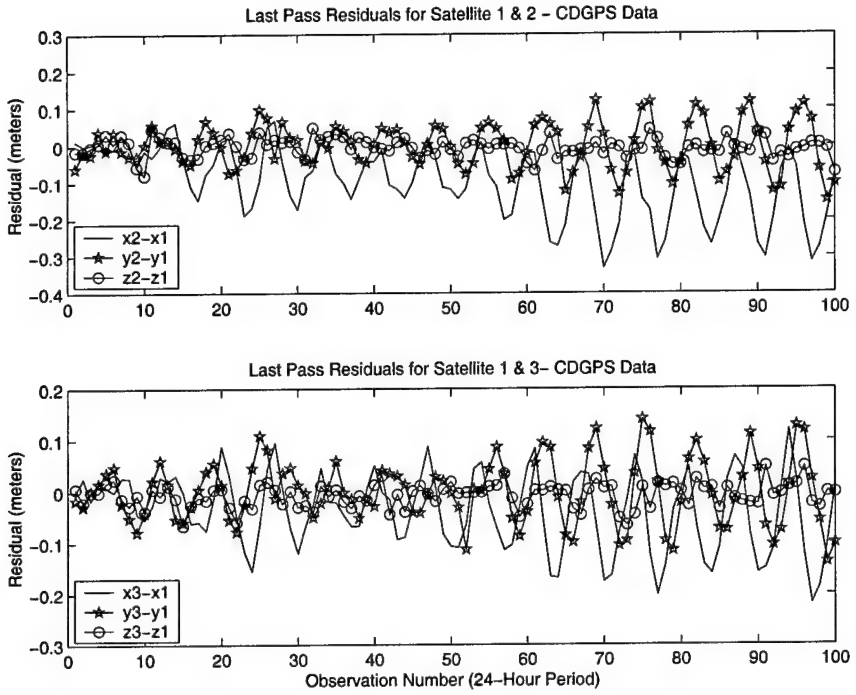

Figure 13 CDGPS Filter's Last Pass Residuals of CDGPS Data (Noise Corrupted Data and Normal Orbital Spacing)

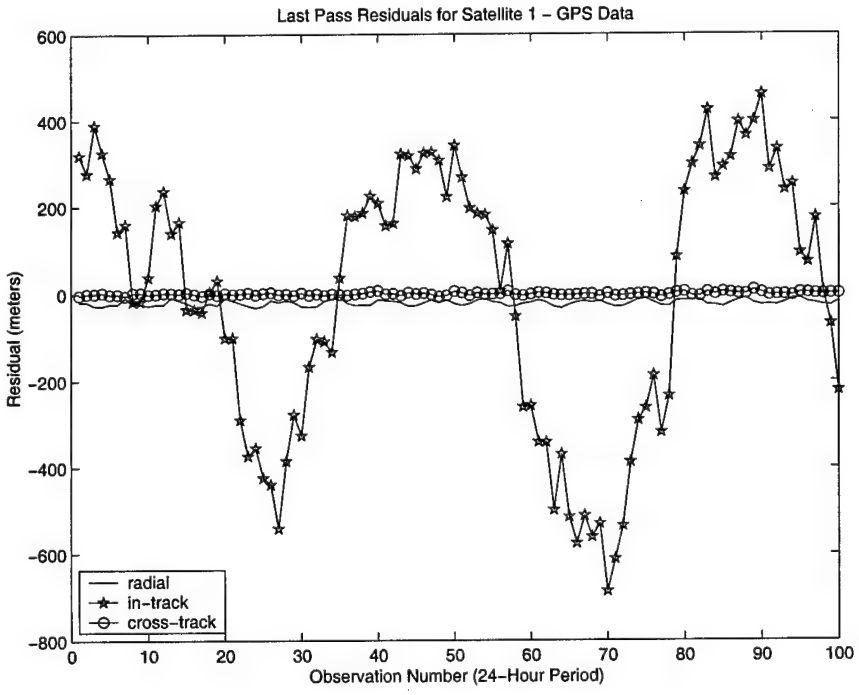

Figure 14 CDGPS Filter's Last Pass Residuals of GPS Data (Noise Corrupted Data and Normal Orbital Spacing)

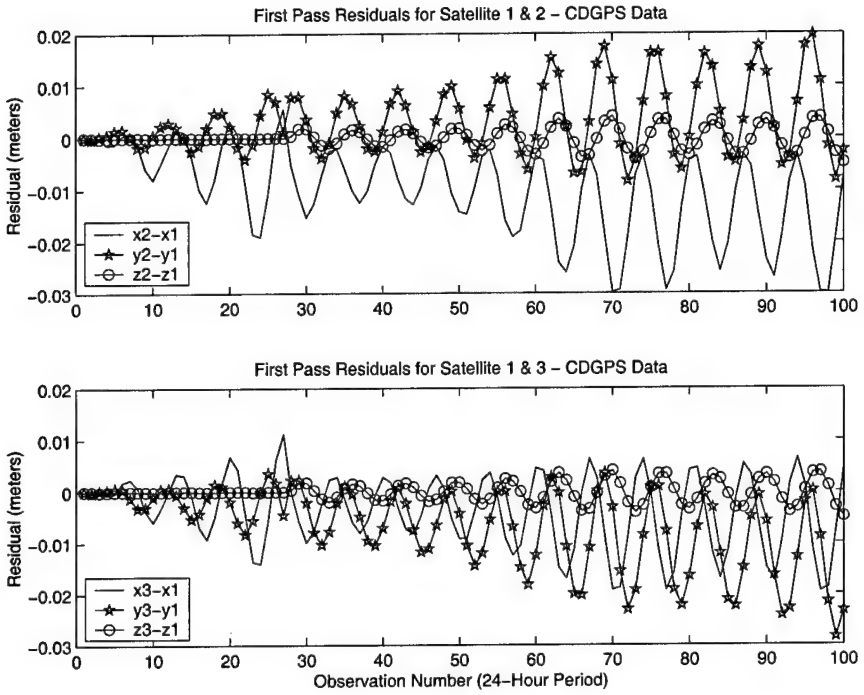

Figure 15 CDGPS Filter's First Pass Residuals of CDGPS Data (Noise Free Data and Reduced Orbital Spacing)

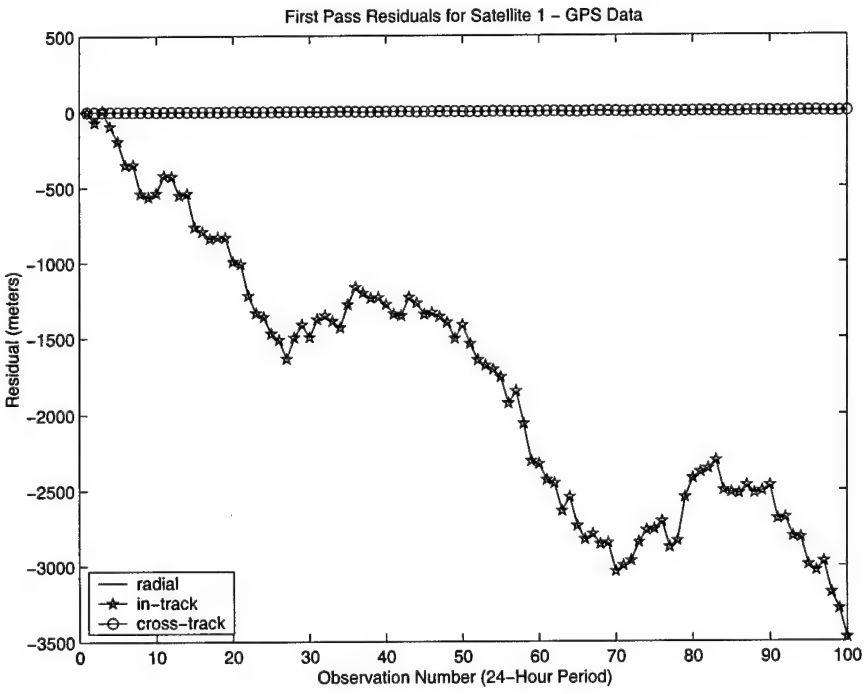

Figure 16 CDGPS Filter's First Pass Residuals of GPS Data (Noise Free Data and Reduced Orbital Spacing)

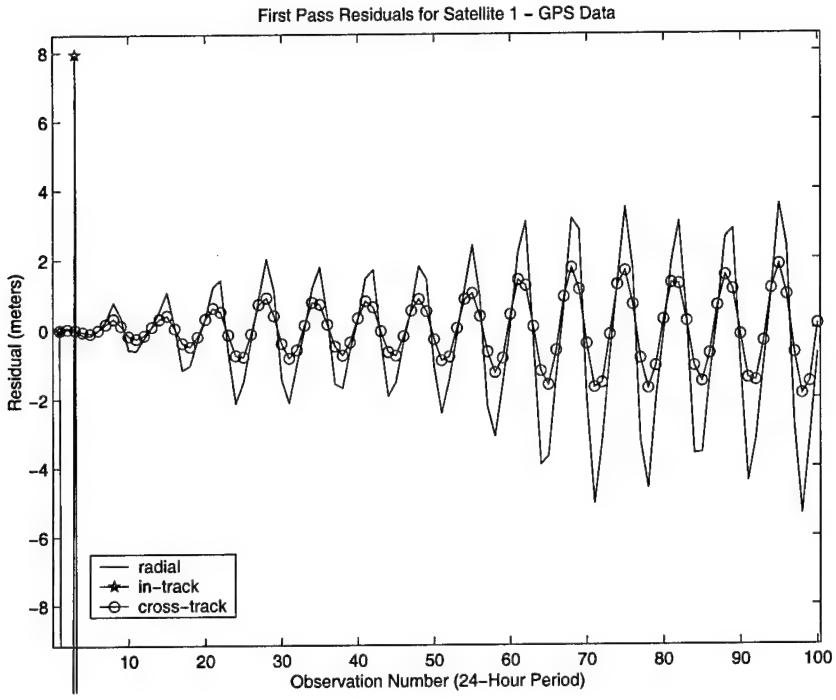

Figure 17 Zoom In of CDGPS Filter's First Pass GPS Residuals (Noise Free Data and Reduced Orbital Spacing)

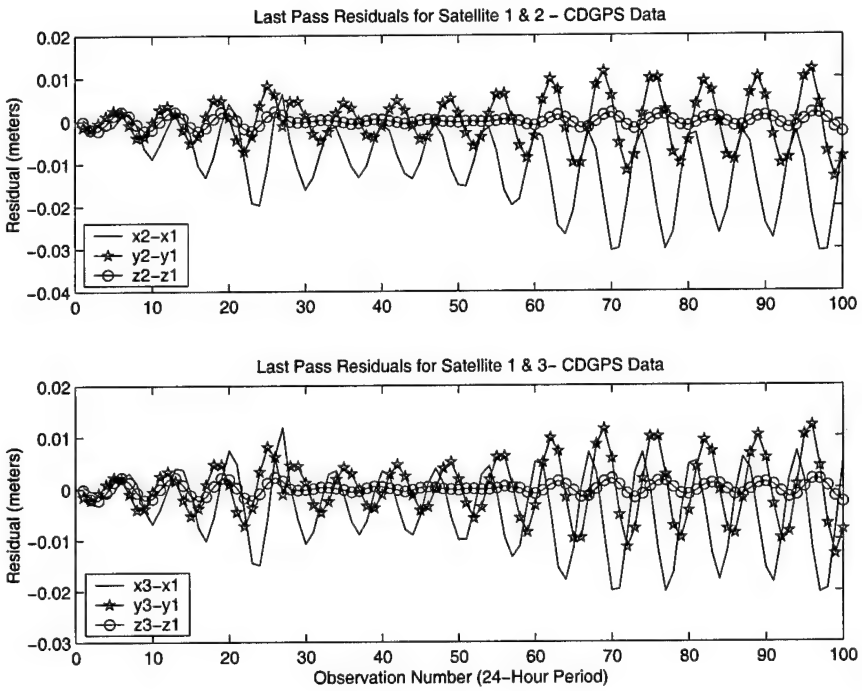

Figure 18 CDGPS Filter's Last Pass Residuals of CDGPS Data (Noise Free Data and Reduced Orbital Spacing)

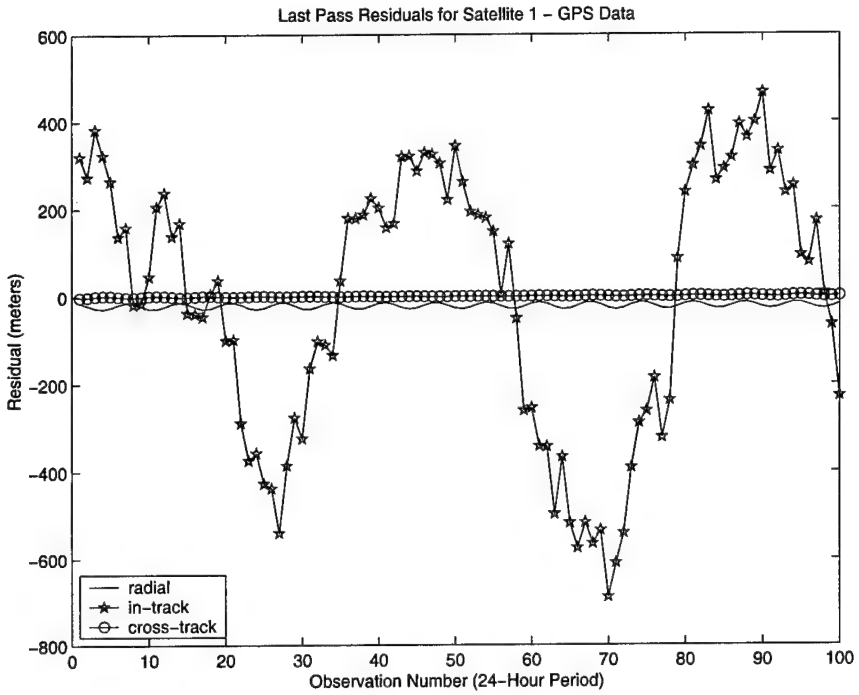

Figure 19 CDGPS Filter's Last Pass Residuals of GPS Data (Noise Free Data and Reduced Orbital Spacing)

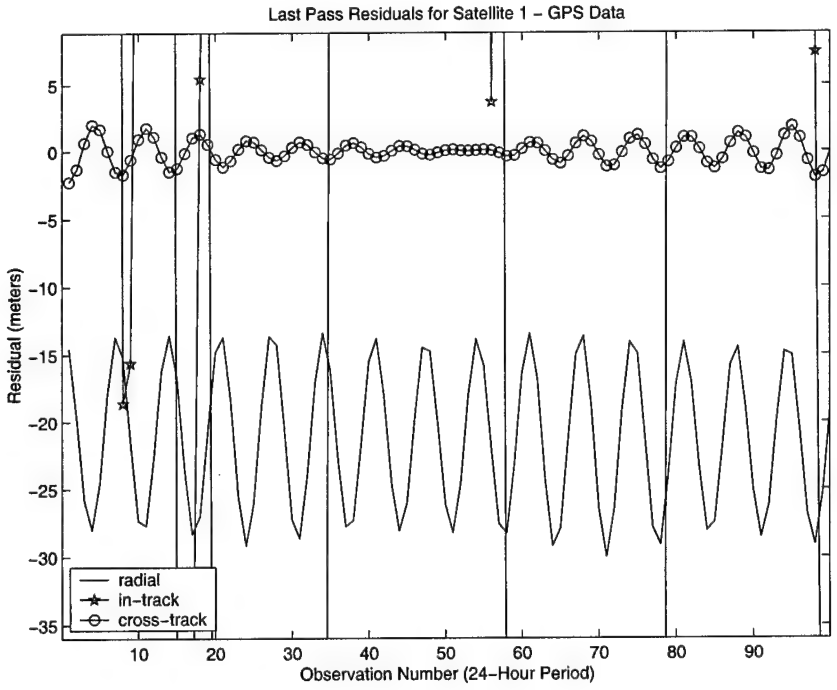

Figure 20 Zoom In of CDGPS Filter's Last Pass GPS Residuals (Noise Free Data and Reduced Orbital Spacing)

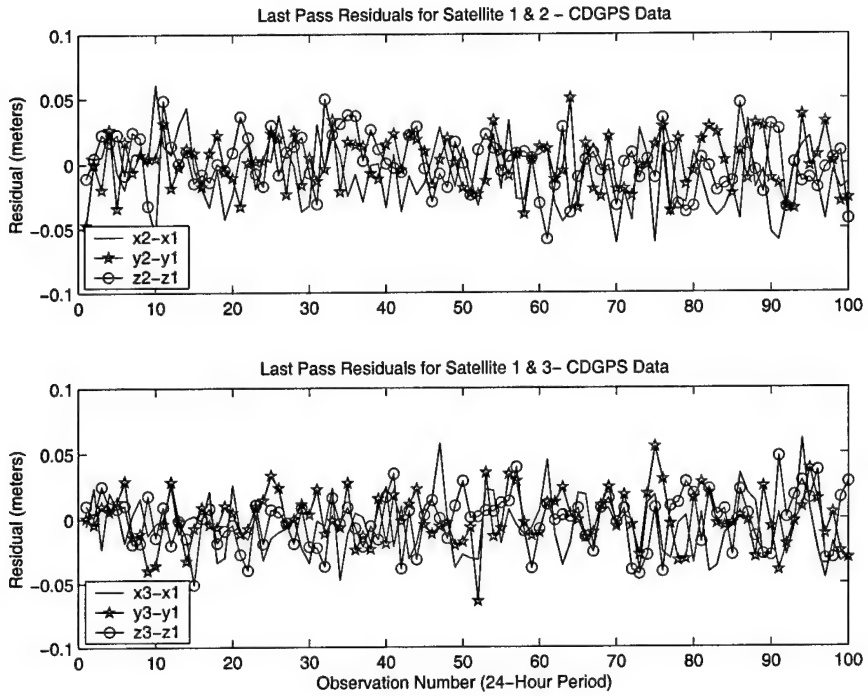

Figure 21 CDGPS Filter's Last Pass Residuals of CDGPS Data (Noise Corrupted Data and Reduced Orbital Spacing)

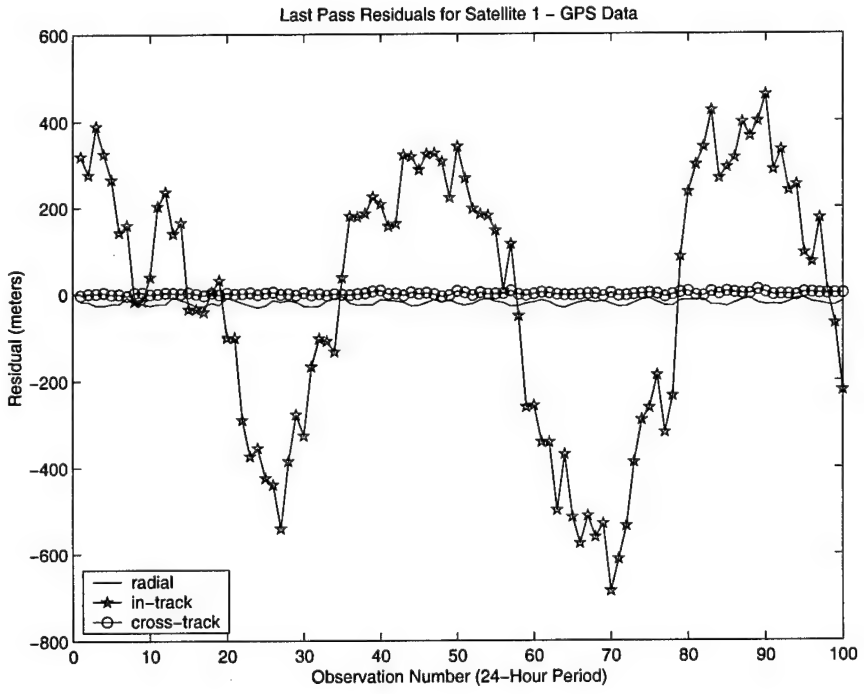

Figure 22 CDGPS Filter's Last Pass Residuals of GPS Data (Noise Corrupted Data and Reduced Orbital Spacing)

4.2 Relative Data Filter - Range Filter

Figure 23 and Figure 24 show the range filter's first and last pass residuals for the noise-free and noise-added runs, respectively. First pass residuals show residuals growing from the centimeter level to nearly 2 meters by the end of 24 hours. The satellite pair comprised of 2 and 3 show a clear, secular growth in their residuals. This growth in the residuals is related to cluster geometry. The effect of formation geometry on residual growth was investigated and will be discussed momentarily. After filter convergence, the last pass residuals show the error bounded within an envelope with maximum magnitudes near 50 meters. The rapid growth of the residuals relative to the other filters' residuals and the size of the last past residuals created uncertainty in the stability of this filter. Singular decomposition analysis was performed on the final covariance matrix to characterize the observability of the filter's states. The analysis showed that while the filter did have 12 significantly large singular values (on the order of 10^5), it also possessed six very small singular values. While they were not exactly zero, their values were near 10^{-3}. This means that while the system gains enough observability from the range measurements to invert the covariance matrix and converge upon a solution, it is still weakly unstable. At this point it is important to recall the form of the range measurements used in this filter. Recalling Equation 50, they are given by

$$\mathbf{Z}_{range} = |\vec{r}_i - \vec{r}_j|, \tag{69}$$

which in the case of three satellites translates to

$$\mathbf{G}_{Range} = \begin{Bmatrix} \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \\ \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2} \\ \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2} \end{Bmatrix}. \tag{70}$$

This shows that the filter is gaining relative position knowledge between the position states of each pair of spacecraft. This is very similar to the what the other relative data filter using CDGPS gains from its measurements. For a three satellite equation, they are described by

$$\mathbf{G}_{DGPS} = \begin{Bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \\ x_3 - x_1 \\ y_3 - y_1 \\ z_3 - z_1 \end{Bmatrix}. \tag{71}$$

While the two measurements are similar in that they both give relative positional information with respect to a satellite pair, there is one very important distinction. The CDGPS measurements do not give as much positional information about the entire formation as the range measurements. The CDGPS measurements are differenced with respect to satellite one. Remember that the assumption of this work is that every satellite within the formation will think it is satellite one (or the central satellite) as it pertains to the estimation problem. The impact of only differencing with respect to satellite one means additional positional information about the relative positions of the other spacecraft with respect to themselves is lost. It is this information that makes the formation in this relative data filter observable and not in the relative data filter using CDGPS measurements. The observability problem in the CDGPS filter was fixed by introducing absolute information in the form of a GPS measurement referenced to the reference periodic orbit. The inclusion of this type of measurement should also make the system more observable in this filter and thus eliminate the weak instability.

Even though the filter is hampered by stability issues, the residuals were still thought to be affected by second order effects. Thus, the relative spacing between the formation elements was reduced by an order of magnitude. The resulting reduced

formation is identical to that of the Relative Data Filter - CDGPS Data. Figures 25 and 26 show the same information as the previous two figures but with the spacing between satellites reduced by an order of magnitude. As hoped, the error bottomed out at the noise floor determined by the measurement covariance programmed into the filter. Although, it can be seen (like before) that the last pass residuals tend to get larger and grow secularly. This can be seen easier when the time scale is expanded. Figure 27 and Figure 28 shows the filter's performance over two and seven days of data, respectively. This appears to be due to second order secular terms in the perturbations of which the filter cannot compensate or due to the stability issues mentioned previously. This trend was examined under different spacecraft geometries to determine how relative positioning affected residual growth.

The formation was reconfigured in phase space (by altering the Floquet modal variables) rather than in physical space since known terms causing secular growth are easier identified and suppressed in phase space. Once the new formation was selected, the physical positions of the spacecraft were determined by transforming them using Equation 46 in Section 3.1.2. Altering the Floquet modal variables must be done with caution so that the initial conditions of the cluster do not cause the cluster to disperse. Specifically, the fourth and sixth modal variable were kept at zero to discourage secular growth in the error and thus cluster dispersion. One data run in particular used an altered formation with the first modal variable of one satellite changed from the initial, reduced cluster design. This modal state was chosen for alteration as it is not inclined to contribute to cluster dispersion. This variable represents the perturbation of the eccentricity and is part of the energy mode. The results from this particular data run can be seen in Figures 29 and 30. The plots still show secular growth like the other cluster, however the satellite pair of 2 and 3 do not linearly diverge like before and the secular growth is less severe. It is interesting to point out that once converged, the residuals of this new formation are actually worse than before. Singular value decomposition was once again performed

on this altered formation's covariance matrix. As implied by the larger residuals, the previously identified small singular values got smaller. This shows that changing the geometry of the formation affects the observability of the system as a whole. While this thesis was not concerned with optimal formation design and thus no more effort was put into observing different formation's behavior, it is recommended for future study.

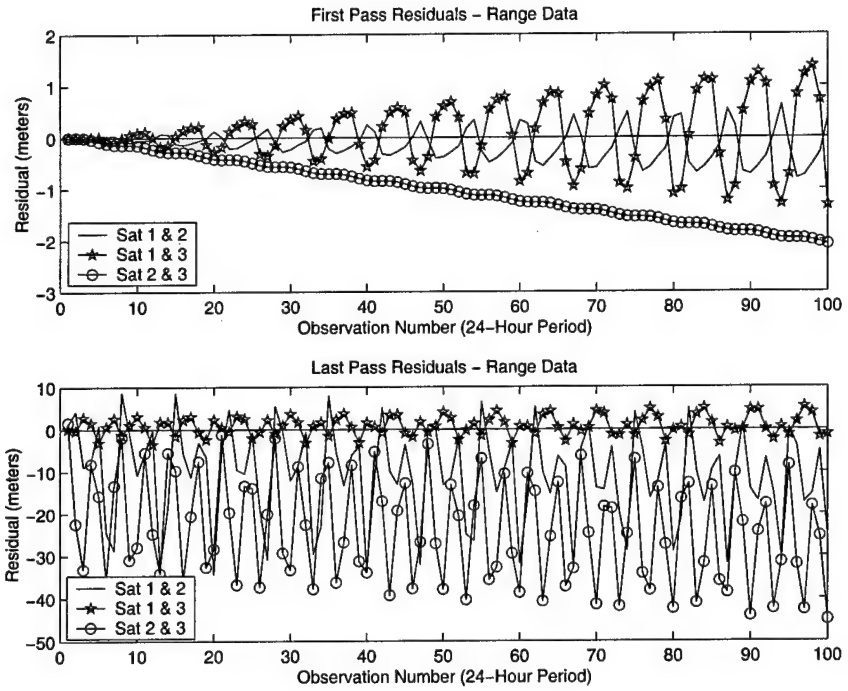

Figure 23 Range Filter's Residuals (Noise Free Data and Normal Orbital Spacing)

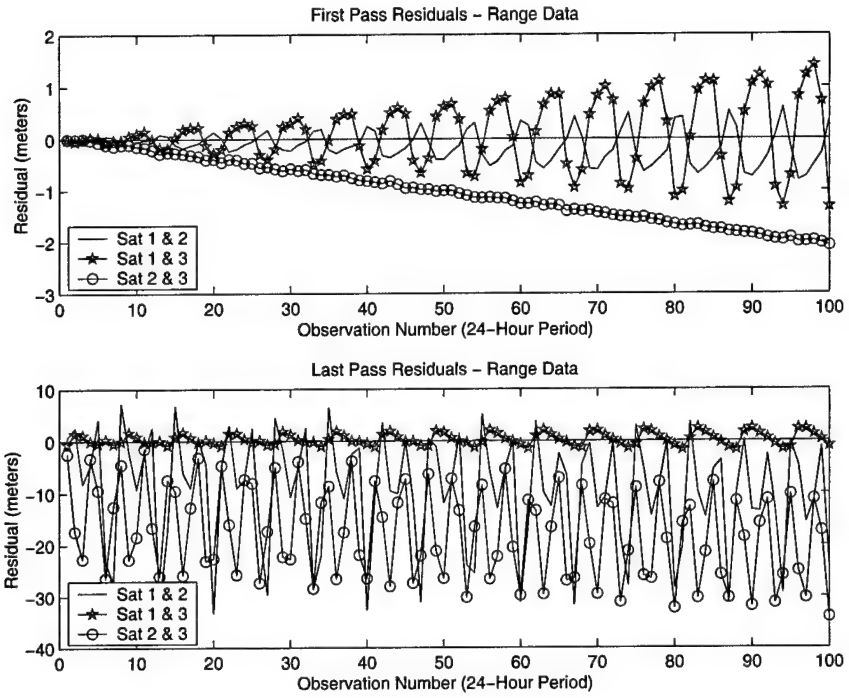

Figure 24 Range Filter's Residuals (Noise Corrupted Data and Normal Orbital Spacing)

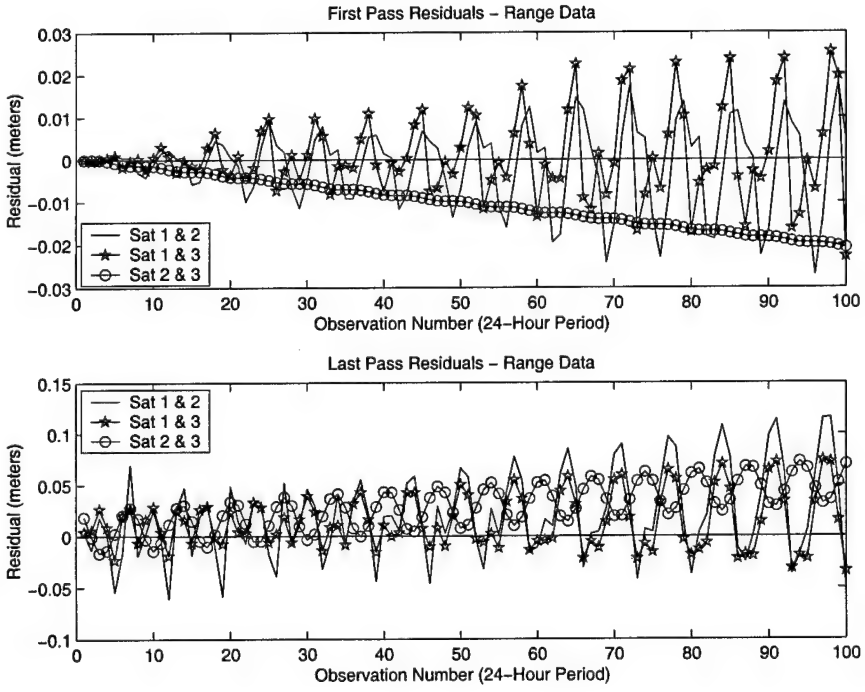

Figure 25 Range Filter's Residuals (Noise Free Data and Reduced Orbital Spacing)

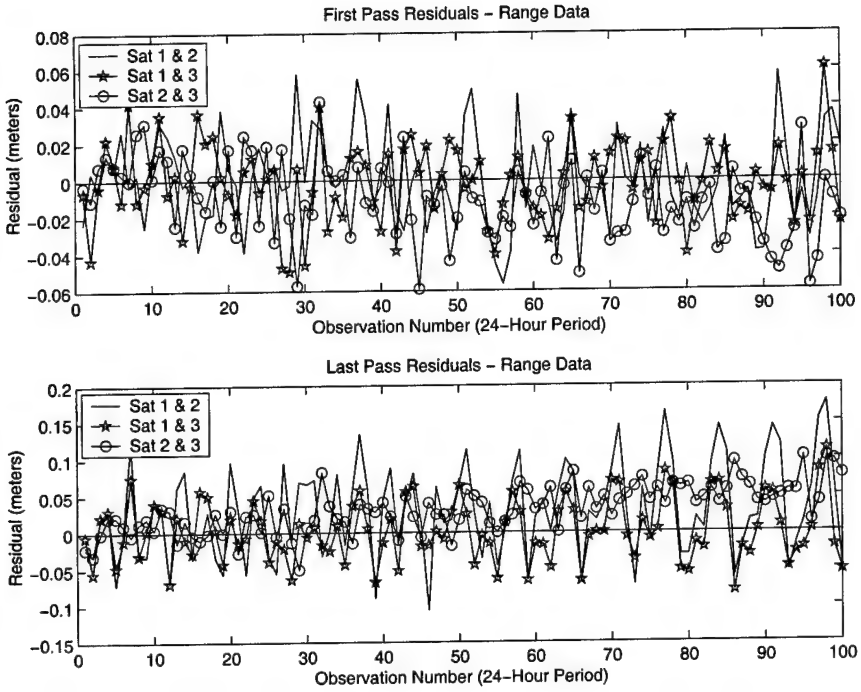

Figure 26 Range Filter's Residuals (Noise Corrupted Data and Reduced Orbital Spacing)

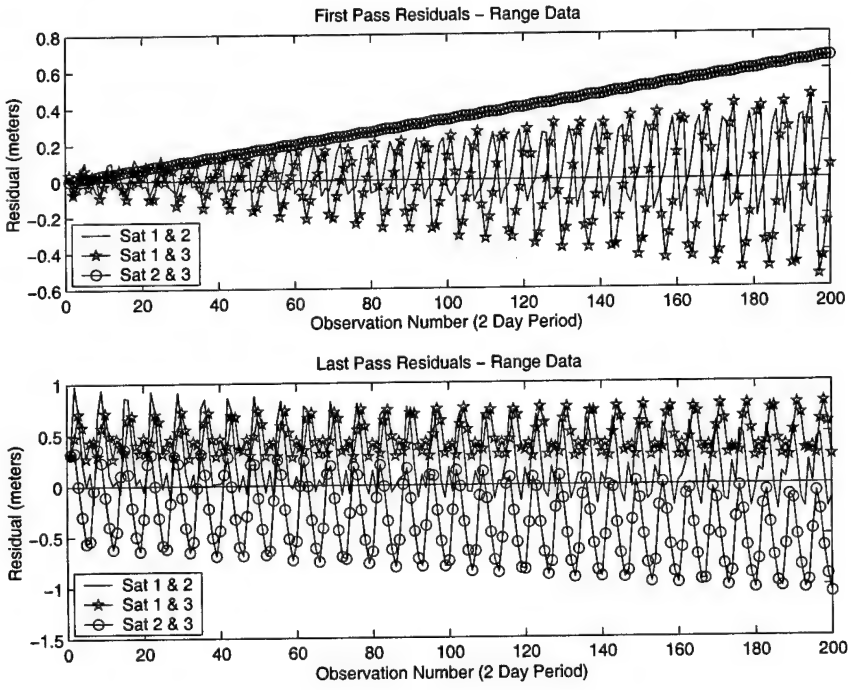

Figure 27 Range Filter's Residuals Over 2 Days (Noise Free Data and Reduced Orbital Spacing)

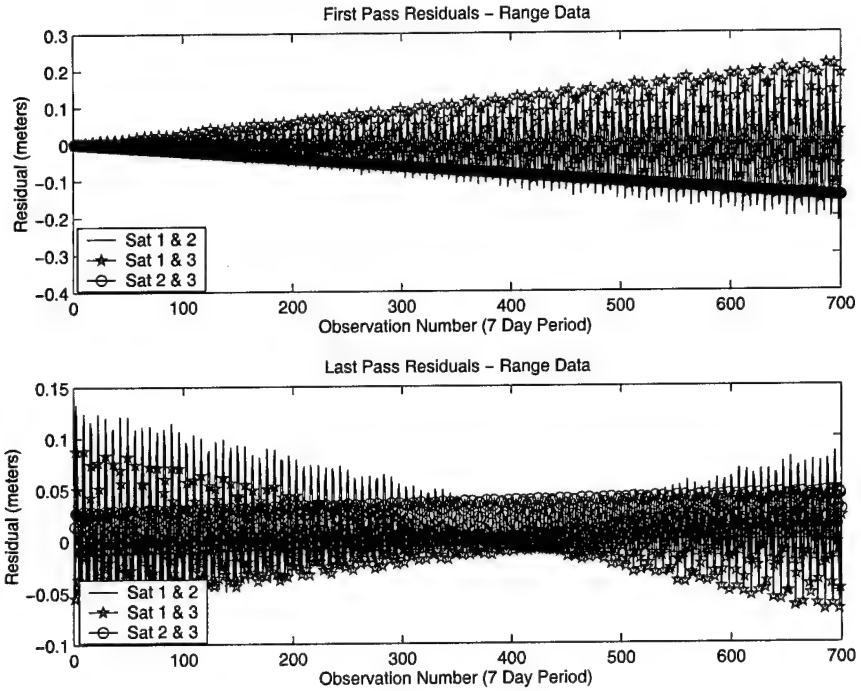

Figure 28 Range Filter's Residuals Over 7 Days (Noise Corrupted Data and Reduced Orbital Spacing)

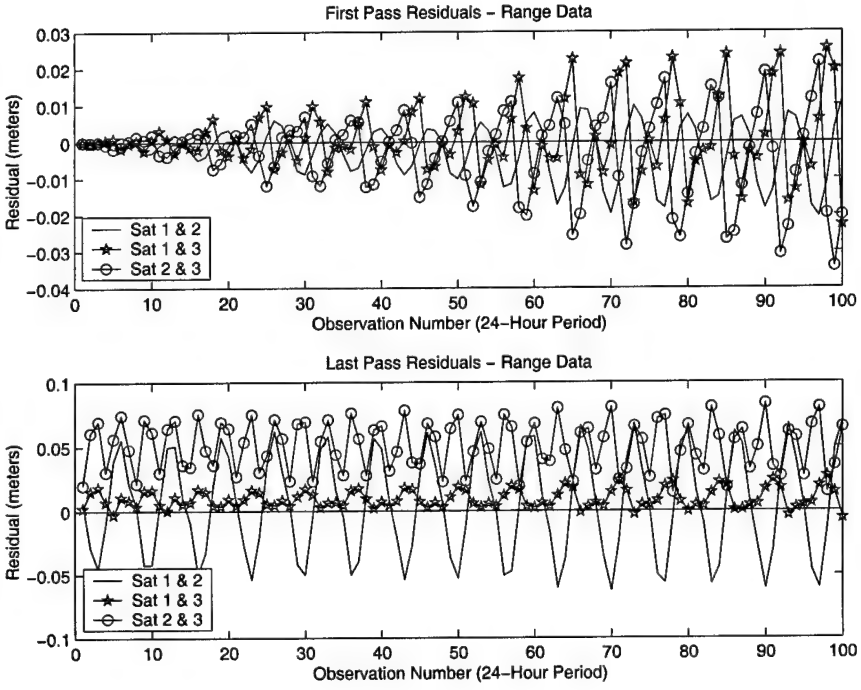

Figure 29 Range Filter's Residuals Over 1 Day (Altered First Floquet Variable From Original Cluster)

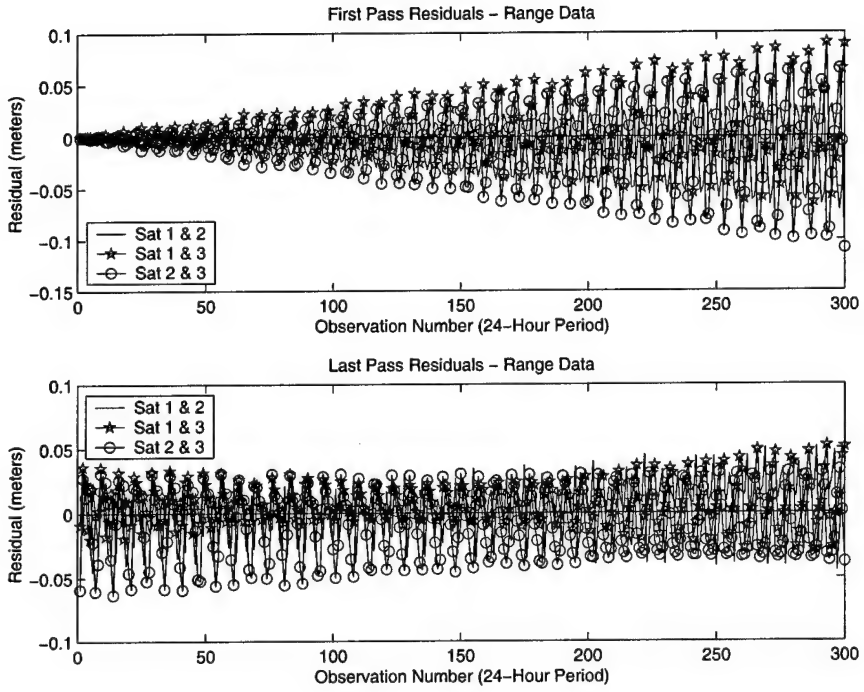

Figure 30 Range Filter's Residuals Over 3 Days (Altered First Floquet Variable From Original Cluster)

4.3 Relative State Filter - CDGPS Data

Figure 31 and Figure 32 show the relative state filter's first and last pass residuals for the noise-free and noise-added runs, respectively. This system is only comprised of two satellites unlike the other two previous filters. Thus, only one satellite pair is represented in the plots. In both figures the performance is similar regardless of the noise added. This once again shows that the errors in the filter's relative motion model are dominating any modeled sensor noise. Similar to the other filter using CDGPS data, the in-track residuals show secular growth. The filter converges and yields residuals with maximum values near 30 centimeters. This error is on the order of the other two filters with the standard, larger spacing.

The spacing was once again reduced by an order of magnitude to lessen the impact of second order effects. Since all three filters use the same formation, this reduced formation is identical to the other filter's reduced formation. The results can be seen in Figure 33 and Figure 34. It is seen that reducing the spacing in the cluster causes the first and last pass residuals to be nearly identical. The with-noise and without-noise measurement plots do not look similar at all as the noise dominates the error in the with-noise plots. A run was accomplished with a poor initial estimate of the system states to verify proper filter functioning. The poor initial estimate was created by randomly adding values of the same order of the states to the states. The results can be seen in Figure 35 and Figure 36. The plots show that the first pass residuals are much larger and that the noise on the measurements no longer dominates the residual trends. The filter does converge correctly and the last pass residuals are identical to those of the runs when the initial state vector estimate was accurate.

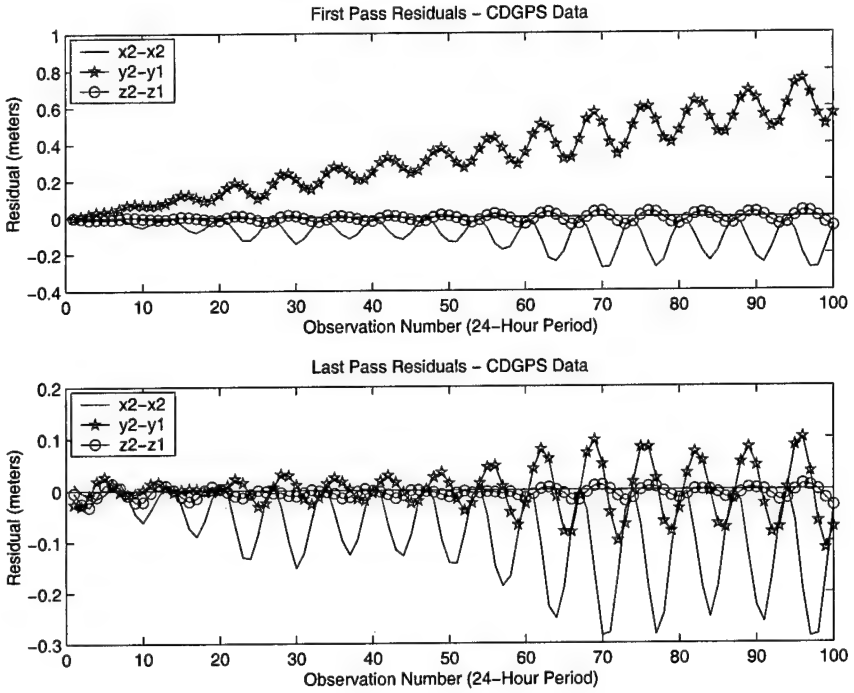

Figure 31 Relative State Filter's Residuals (Noise Free Data and Normal Orbital Spacing)

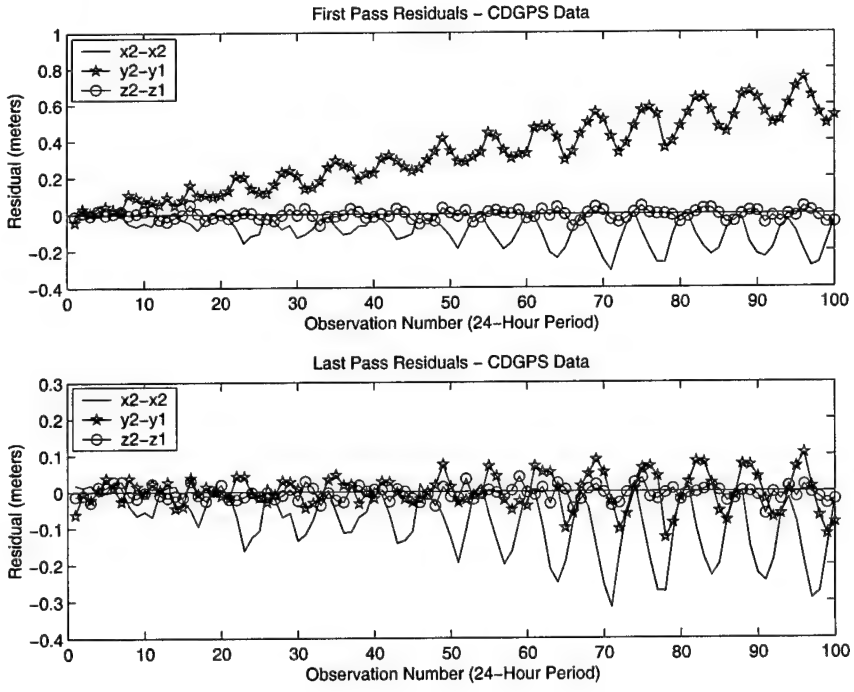

Figure 32 Relative State Filter's Residuals (Noise Corrupted Data and Normal Orbital Spacing)

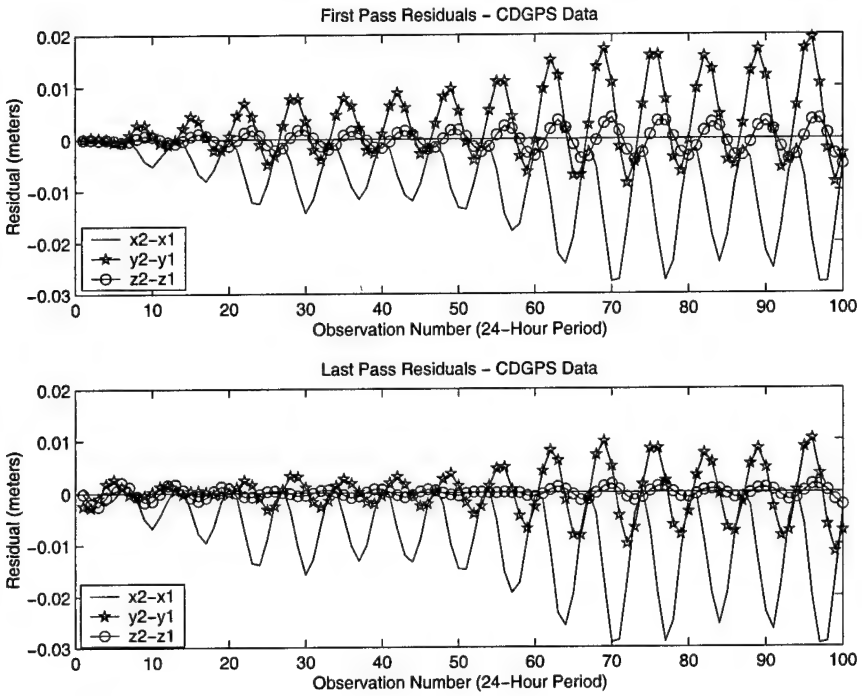

Figure 33 Relative State Filter's Residuals (Noise Free Data and Reduced Orbital Spacing)

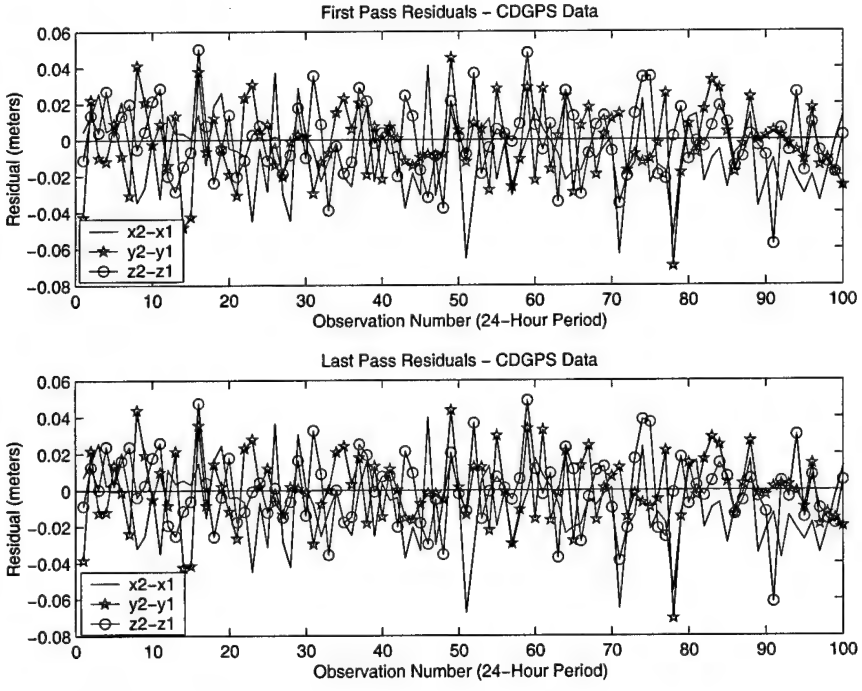

Figure 34 Relative State Filter's Residuals (Noise Corrupted Data and Reduced Orbital Spacing)

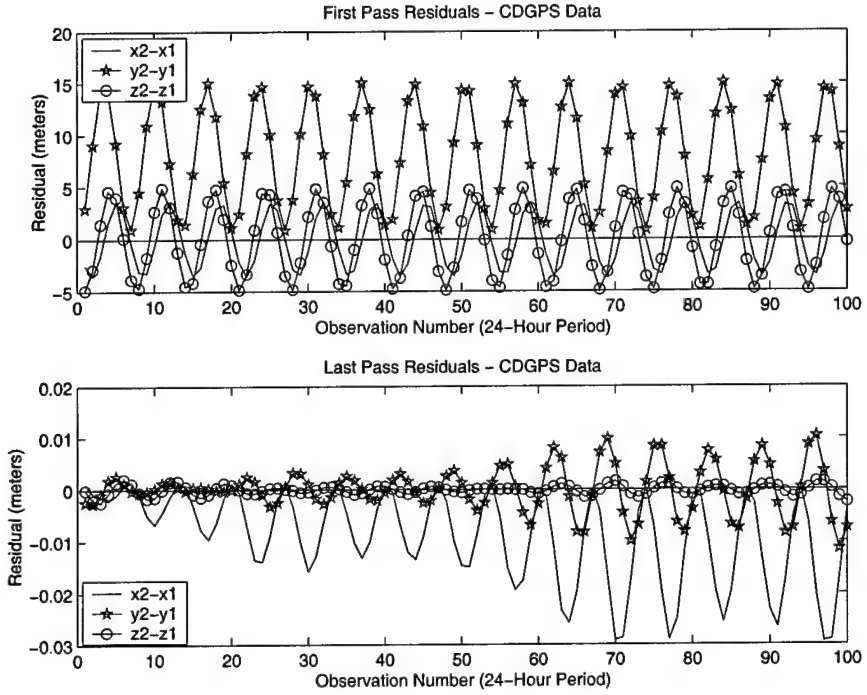

Figure 35 Relative State Filter's Residuals (Noise Free Data, Poor Initial State Estimate and Reduced Orbital Spacing)

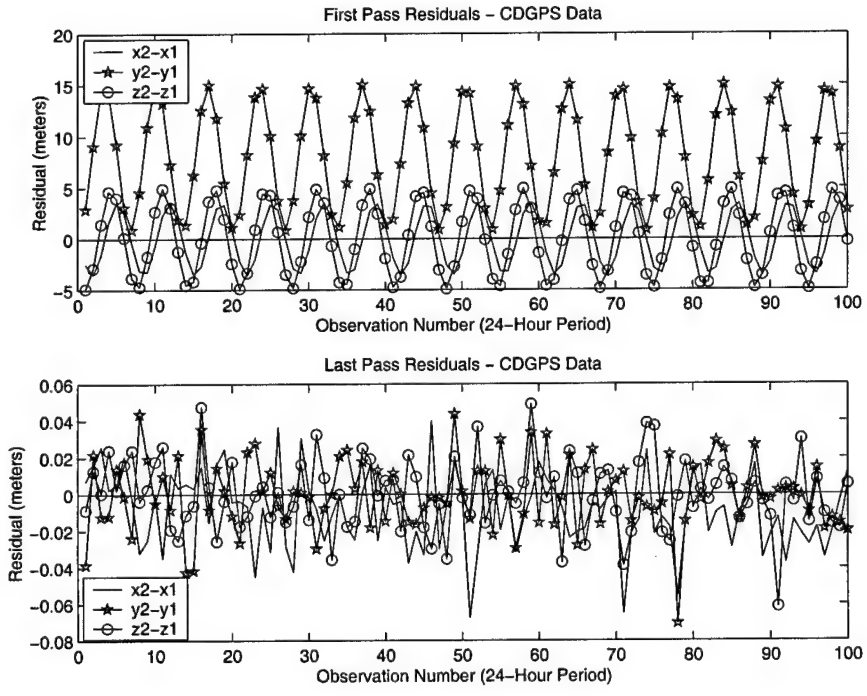

Figure 36 Relative State Filter's Residuals (Noise Corrupted Data, Poor Initial State Estimate and Reduced Orbital Spacing)

4.4 *Second Order Effects*

The previous sections have shown that the limiting factor in the implementation of this filter is the effect of the second order dynamics. If one were to design a satellite cluster with spacing on the order of tens of meters as opposed to kilometers, this filter would suffice. However, as the data has shown, the residuals even with perfect measurements grow unacceptably large. The second order effects can be compensated for by including the higher order terms up to the second order during linearization via Taylor Series expansion. For each satellite, this can be accomplished via

$$\dot{\mathbf{x}} = A_\alpha(t)\mathbf{x} + \frac{1}{2!}B_{\alpha\beta}(t)\mathbf{x}_\alpha\mathbf{x}_\beta + \ldots + \mathbf{f_p}, \tag{72}$$

where the Greek indices are summed one through six since there are six states for each satellite. Thus, the quantities of $A_\alpha(t)$ and $B_{\alpha\beta}(t)$ are periodic tensors formed by taking partial derivatives of the Hamiltonian function. This can be expressed by

$$\begin{aligned} A_{ij}(t) &= X_{i\alpha}\frac{\partial^2\mathcal{H}'}{\partial\mathcal{X}_\alpha\partial\mathcal{X}_j}\bigg|_{X_0(t)}, \\ B_{ijk}(t) &= X_{i\alpha}\frac{\partial^3\mathcal{H}'}{\partial\mathcal{X}_\alpha\partial\mathcal{X}_j\partial\mathcal{X}_k}\bigg|_{X_0(t)}, \end{aligned} \tag{73}$$

where $X_{i\alpha}$ is a matrix of off-diagonal values of +/-1 required to produce the structure of Hamilton's equations of motion [35]. This is not much more work than is already accomplished since $B_{\alpha\beta}(t)$ is only one more round of partial differentiation past what is already done to get $A_\alpha(t)$. Additionally, since $A_\alpha(t)$is mostly zeroes or constants, $B_{\alpha\beta}(t)$ will mostly be populated by zeroes and only require a few derivatives to be taken.

4.5 *Relative vs. Absolute Position Estimation*

One of the objectives of this research was to separate the relative positioning problem from the absolute positioning problem in the navigation of a satellite for-

mation. The data clearly shows that this has been achieved. Past efforts have tried to obtain centimeter level estimates in the relative positions by obtaining centimeter level estimates in the absolute positions. This approach had miserable results as a majority of the significant digits were wasted in estimating the absolute positions. This research, using mostly relative data, produced absolute position estimates that, although may be adequate (depending on the application), are relatively poor. However, the relative position estimates generated by this filter exhibit error that is directly proportional to the error in the measurements. Thus, the accuracy of the filter's performance is limited to the type of data used, centimeters in this work. This extremely accurate positioning is obtained regardless of the performance of absolute positioning. This separation between the relative and absolute positioning problem is seen in the residual plots as well as in a closer inspection of the system's covariance matrix.

The covariance for the CDGPS (and GPS) filter is an 18 by 18 matrix and is essentially comprised of nine smaller six by six covariance matrices (referred to here as covariance sub-blocks). If one views the state vector as one large vector comprised of three smaller state vectors for each satellite, it is easier to see this sectioned description of the system's covariance matrix. Three of the covariance sub-blocks correspond to each satellite and six correspond to each satellite pair. Since there are only 3 unique satellite pairings in a three satellite system, there are two copies of the same covariance sub-blocks for each pairing. The system's covariance matrix and its nine sub-blocks are represented by Equation 74. It can be seen that the off-diagonal blocks pertain to satellite pairs (relative positioning) while the blocks along the diagonal relate to the state of each satellite (absolute positioning).

$$P_{system} = \begin{pmatrix} P_{x_1} & P_{x_1-x2} & P_{x_1-x_3} \\ P_{x_2-x_1} & P_{x2} & P_{x_2-x_3} \\ P_{x_3-x_1} & P_{x_3-x2} & P_{x_3} \end{pmatrix} \tag{74}$$

The off-diagonal blocks of the covariance can be defined as

$$P_{x_i - x_j} = E\left((\delta x_i - \delta x_j)(\delta x_i - \delta x_j)^T\right), \tag{75}$$

where δx is the true error in the physical states and i and j range from one to the number of satellites in the problem. Recalling from Section 3.1.2 that the physical states, x, are transformed from the modal states, z, via a the transformation matrix F, the true error can be expressed as

$$\delta x_i = F \delta z_i. \tag{76}$$

Using this relationship and the fact that the modal matrix, F, is deterministic, Equation 75 becomes

$$\begin{aligned} P_{x_i - x_j} &= FE\left((\delta z_i - \delta z_j)(\delta z_i - \delta z_j)^T\right) F^T \\ &= F\left(E(\delta z_i \delta z_i^T) - E(\delta z_i \delta z_i^T) - E(\delta z_i \delta z_i^T) + E(\delta z_i \delta z_i^T)\right) F^T \\ &= F\left(P_{z_i} - P_{z_i} - P_{z_i} + P_{z_i}\right) F^T. \end{aligned} \tag{77}$$

The blocks along the diagonal of the covariance are similarly found to be

$$\begin{aligned} P_{x_i} &= E(\delta x_i \delta x_i^T) \\ &= E\left(F \delta z_i (F \delta z_i)^T\right) \\ &= FE\left(\delta z_i \delta z_i^T\right) F^T \\ &= F P_{z_i} F^T. \end{aligned} \tag{78}$$

The above relationships were investigated within the covariance matrices from filter runs using both relative and absolute GPS measurements. The results of this effort showed a distinct boundary between the two data types. While the filter covariance was pessimistic, the covariance values for the relative data were still between 10 and

30 times smaller than the magnitudes of the absolute data. The pessimism in the filter can be attributed to the lack of second order dynamics in the filter's dynamics model, the reference periodic orbit mismatch, and the absence of any effort to tune the filter.

V. Conclusions and Recommendations

This work shows that it is possible to estimate relative satellite positions using recursive Bayesian estimation techniques with relative measurement data as the primary measurement type. Each of the three filters studied converged with errors on the order of the measurement noise used when the orbital spacing is on the order of tens of meters. However, the filter yielded errors an order of magnitude higher when the orbital spacing was increased to a much more desirable distance for space-based radar applications, on the order several hundred meters. Despite the fact that the two relative data filters have been shown to require absolute data to ensure system observability, the most important objective of this research has been accomplished. This is the separation of the relative positioning problem from that of the absolute. Even with extremely poor absolute position knowledge, the filter was able to converge on a relative solution with exquisite accuracy. In fact, it has been shown through this work that the relative motion model inaccuracies are the dominating noise source and not the sensor noise added to the measurements.

Results show that the filter models the cluster diverging as a unit for perturbations that effect the group as a whole. These perturbations could include air drag, gravity harmonics, solar flares, etc. The absolute positioning of the cluster may suffer under these perturbations, but this does not matter much since centimeter level accuracy in the estimate of the system's absolute position is not required for most applications. Accuracy on the order of tens of meters should be more than enough to perform uplink and downlink communications or reconfiguration maneuvers. On the other hand, centimeter or better knowledge of the relative spacecraft positions is imperative in applications such as space-based radar where accuracy is needed on the order of a tenth of the wavelength used. Thus, the fact that the filter shows the cluster degrading as a unit is encouraging to the future of this type of application. Also encouraging is the level of accuracy achieved, not only for the implementation

of radar but also for maintaining accurate CDGPS estimates. Since the filter is showing performance on the order of centimeters, this accurate position knowledge can be used by a spacecraft's on-board CDGPS system to resolve the unknown integer ambiguity experienced when a CDGPS system loses lock on GPS signals. Since the system dynamics are very deterministic (and the filter has a position estimate accurate to within a few centimeters), the filter can trust its dynamics model's position predictions during a loss of GPS signal lock to help resolve the integer ambiguity and quickly regain lock. This is true if the error in the position is less than the length of a GPS wavelength and of course the signal outage does not last any significant amount of time. The GPS wavelengths are about 19 centimeters for L1 and 24 centimeters for L2 (L1 and L2 are the downlink frequencies for user data). While the error shown in this work is not much lower than the GPS wavelengths, the accuracy obtained is still encouraging since GPS carrier-phase filtering algorithms do not usually use L1 or L2 wavelengths. Instead they use an observable comprised of a combination of the wavelengths, called a measurement combination, since L1 and L2 wavelengths are relatively small. Measurement combinations essentially lengthen the effective wavelengths of GPS signals and this added length helps speed up the resolution of the integer wavelength ambiguity. In particular, one observable called a Widelane measurement has a effective wavelength of 86 cm. This is much larger than the error shown in this work and thus the accurate position estimates will speed the resolution of the integer ambiguity, or at least help provide a better guess at the ambiguity, when GPS signals are not being properly tracked.

Since this work can be described as a proof of concept, future work should focus on improving the initial results obtained. The areas for improvement include adding in perturbations, second order dynamics, GPS measurements to the relative data filter using range data, and using a fitted reference periodic orbit. The easiest recommendation to implement will be the addition of the GPS measurements to the range measurement filter. As mentioned previously, this should make the system

more observable and improve the filter's perfomance. The addition of second order dynamics should prove to be a little more difficult to implement. The decision to not include second order system dynamics in the filter has been identified as the reason behind each filter's poor relative position performance when the size of the cluster is of several hundred meters. Further work should involve the inclusion of second order terms in the linearization of the system dynamics to allow the size of the cluster to be expanded. Once this issue is addressed, the cluster can be expanded to determine the maximum spacing a cluster may have before the measurement accuracies are degraded due to other higher order effects (third order and higher). While second order effects are affecting the relative residuals, the absolute residuals (and the relative to a smaller extent) are being affected by a reference periodic orbit mismatch. It is recommended that future work use a fitted reference periodic orbit to remove the periodic oscillations from both the absolute and relative residuals as well as improve the overall accuracy of the absolute estimates. This can be accomplished by running a periodic orbit estimator in concert with the positional estimator. The last significant effort from an improvement perspective should be to model the effects of perturbations. The effects of air drag, and tesseral and sectoral harmonics should be modeled as a forced solution within the linear dynamics of the filter and the effects investigated. Although this work demonstrated accurate estimation of relative positions in the absence of perturbations, it is believed that if the relative spacing of the formation is not too large, the perturbating forces will cause the cluster to be perturbed as a unit and not disperse. Thus, accuracies with perturbations should be close to the accuracies obtained in this effort without them.

Once these four improvement efforts have been accomplished, it is recommended that implementation issues such as filter tuning, size of the data batch to process, and trade-offs in the filter algorithm chosen be investigated. The filter should be tuned over many data runs through a Monte Carlo analysis, adjustment of the noise covariance values, or a combination of the two. This will make the co-

variance output with every estimate more inline with the estimates provided, unlike the pessimistic covariance of this work. Of course prior to any tuning efforts, the proper length of the data batch interval should be determined so that trends due to modeled perturbations can be fully characterized. Although more data is usually better, especially when trends in the residuals due to certain perturbations span over several days, the batch size may need to be truncated to save time and processing power since both of these quantities are not luxuries in a real-time space application. Lastly, the Bayes algorithm should be compared against other filtering algorithms, such as the Kalman filter, to see the advantages (and disadvantages) of different approaches. Although this author believes that any successful filtering approach will require a batch formulation, it may be worthwhile to observe the difference in filter performance from a non-batch formulation as well.

In conclusion, this research has two major successes. First, it has been shown that relative spacecraft position estimates can be identified to within the accuracy of the measurements used, potentially on the order of centimeters, if the spacing between the cluster elements is kept on the order of meters (larger distances if second order dynamics are included in the filter). The second major success is probably the most important. The relative positioning problem has successfully been detached from the absolute positioning problem. Extremely accurate estimates of relative spacecraft positions can be made even though knowledge of the absolute positions are poor.

Appendix A. Relative Data Filter Source Code

```
//---------------------------------------------------------------------------
//  Non-Linear Bayes Algorithm Used to Estimate Satellite Positions in
//  the Orbital Frame, or Clohessy-Wiltshire Frame (Radial, Intrack,
//  Cross-track).  This code represents the relative data filters and not the
//  relative state filter.  The relative state filter has minor modifications
//  that are described in the thesis text.
//
//  Created by Captain Ralph Bordner III
//  17 October 2000
//  Original Version
//  Last revision done on 15 February 2001.
//
//  This code is written mostly in the C language with a few bits of C++
//  thrown in as well.  It was written for Borland's C++ Builder 5.  The code
//  was executed on a PIII - 533 MHz system with 128 MB of RAM.  The OS was
//  Winodws 98 and Windows ME.
//
//  Portions of this code were translated from FORTRAN code written
//  by Dr. William Wiesel, Professor of Astronautical Engineering, AFIT
//

#include <vcl> #pragma hdrstop #include <iostream> #include
<fstream> #include <cstdlib> #include <cmath> #include <conio>
#include <math>

using namespace std;

//---------------------------------------------------------------------------
//  Declaration of Additional Funtions Used
//---------------------------------------------------------------------------

void dynamics(long double phi[][18], long double xpred[],
```

```
            long double &tob, long double modal_ref[],
            long double &phase_angle, int &num_sats,
            long double modal_pred[]);
void obser_processor(long double &tob, int &iztype, long double
Qinv[][18],
                     long double zpred[], long double h[][18],
                     long double xpred[], int &num_sats);
void matrix_inverter(long double a[][18], long double ainv[][18],
int &ier,
                     long double &condnum, int &num_sats);
void soln1a(long double &t, long double &phi, long double Xpo[],
          long double F[][6], long double expJt[][6]);

//---------------------------------------------------------------------------
//  Main Routine
//---------------------------------------------------------------------------

#pragma argsused
      int main(int argc, char* argv[])
{
        //-------------------------------------------------------------------
        //     Declarations and Initializations
        //-------------------------------------------------------------------
        //  Declare the in stream 'in' and 'data' from class ifstream (C++).
        //  Declare the out file 'out' and 'input' the regular C way so that
        //  file output can be formatted easier.

        ifstream in, data;     // 'in' is the initial state and 'data' is data.
        FILE *out, *input;     // 'out' is Bayes output and 'input' is the
        FILE *sample, *resids; // 'in' file for the next iteration of Bayes.
        FILE *modal_corr, *modal_est;
        FILE *resids_size, *oneday, *phys, *modal, *meas, *type;

        long double modal_minus[18], pinv_minus[18][18], epoch_time, reject;
```

```
long double modal_ref[18], observation[9][1000], time[1000], z[6];
long double tmat[18][18], htq1[18][6], phase[1000], phase_angle, tob;
long double condnum, h[6][18], xpred[18], Qinv[18][18], zpred[18];
long double sqroot, residuals[20][1000][6], htq1r[18], pminus[18][18];
long double p[18][18], phi[18][18], phip[18][18], diff_modal[20][18];
long double pinv[18][18], phi_inv[18][18], modal_ref_mat[20][18];
long double converted_resid, std_dev, sps_gps_sigma;
long double physical_states[20][1000][18], modal_pred[18];
long double modal_states[20][1000][18];

int maxit, i, j, nob, iter, iob, ier, datatype[1000], numdata[1000];
int irej, k, num, iztype, rej_flag[20][1000][6], max_iter, flag;
int size_state, num_sats, max, egm_variable, tu, data_points;
int int_step, output_type, seed, sectoral_flag, tesseral_flag;

bool done, converge, finished, eof();

//**********************************************************************
//  Debug code to print out covaiance.
//if((sample=fopen
//     ("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\sample.out",
//     "w"))==NULL){
//     printf("Cannot open file.\n");
//}
//**********************************************************************
//  Hard-coded state size - can be rewritten to be brought in via
//  data file.

num_sats = 3;
size_state = num_sats*6;

//  Initialize the residual array based on dimensions declared above.

for(i=0; i<20; i++){
```

```
     for(j=0; j<3000; j++){
          for(k=0; k<6; k++){
               residuals[i][j][k] = 0.0e+00;
          }
     }
}

//-----------------------------------------------------------------
//     Read In Initial Values
//-----------------------------------------------------------------
//  Open the input file from working folder and
//  read in initial estimates.

in.open("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\state.tf",
        ios::in);

//  Error handling in case file does not open.

if(!in.is_open()){
     cout << "The input file for initialization could not be opened.\n";
}

//  Read in epoch time.

in >> epoch_time;

//  Read in initial guess.

for(i=0; i<size_state; i++){
     in >> modal_minus[i];
}

//  Read in value of maximum allowed iterations.
```

```
in >> maxit;

//  Read in residual rejection criteria (in sigma).

in >> reject;

//  Read in the Inverse Covariance.

for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          in >> pinv_minus[i][j];
     }
}

//  Set the first guess to the previous estimate.

for(i=0; i<size_state; i++){
     modal_ref[i] = modal_minus[i];
}

for(i=0; i<size_state; i++){
     modal_ref_mat[0][i] = modal_ref[i];
}

//  Close the 'in' stream.

in.close();
//-----------------------------------------------------------------------
//     Read In Data
//-----------------------------------------------------------------------
// Open observation file and read in data.

data.open("c:\\windows\\desktop\\thesis\\fromwiesel\\truthmodel\\data.in",
          ios::in);
```

```
// Error handling in case observation file does not open.

if(!data.is_open()){
     cout << "The obseration file could not be opened.\n";
}

// Read in observation data.

i=0;
while(i<1000){
     // Read in data type amd time of data sample.

     data.eof();

     if(data.eof() == true){
          cout << "End of Data File Reached.\n";
          break;
     }

     data >> time[i] >> datatype[i] >> phase[i];

     // Read in measurements depending on type.

     if(datatype[i]==1){ //  GPS data - relative
          numdata[i] = (num_sats-1)*3;
          num = numdata[i];
          for(j=0; j<num; j++){
               data >> observation[j][i];
          }
     }
     else if(datatype[i]==2){  //  Range data
          numdata[i] = ((num_sats*num_sats-num_sats)/2);
          num = numdata[i];
```

```
            for(j=0; j<num; j++){
                data >> observation[j][i];
            }
        }
        else{   //  GPS data with absolute flavor
            numdata[i] = 3;
            num = numdata[i];
            for(j=0; j<num; j++){
                data >> observation[j][i];
            }
        }

        // Determine if storage array for data is full.

        if(i==2999){
            cout << "Obsevation buffer full...truncated.\n";
        }

        // Increment counter and determine number of measurements read.


        nob = i;
        i = i + 1;
}

//  Close the 'data' stream.

data.close();

//  Open output file.

if((out=fopen
   ("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\bayes.out",
    "w"))==NULL){
```

```
     printf("Cannot open output file.\n");
}

fprintf(out,
"***Output from Bayes Filter for Relative Satellite Navigation***\n");
fprintf(out,
"(Output in DU (1678.135 km = 1 DU) unless otherwise stated\n\n");
//  Output initial state
fprintf(out, "\nInitial state vector:\n\n");
     for(i=0; i<size_state; i++){
          fprintf(out, "%+.15Le\n", modal_ref[i]);
}
fprintf(out, "\n");

// Set last pass flag to false for processing loop that follows.

finished = false;

//-------------------------------------------------------------------
//      Begin Iteration Loop - Non-Linear Least Squares
//-------------------------------------------------------------------

for(iter=1; iter<(maxit+1); iter++){

     //---------------------------------------------------------------
     //   Initialize Buffers for Matrix Product Accumulation
     //---------------------------------------------------------------
     // Initialize htq1r to pminus*(xminus-xref).
     // Initialize pinv to pminus.

     for(i=0; i<size_state; i++){
          htq1r[i] = 0.0e+00;
          for(j=0; j<size_state; j++){
               pinv[i][j] = pinv_minus[i][j];
```

```
            htq1r[i] = htq1r[i] + pinv_minus[i][j]*(modal_minus[j]
            - modal_ref[j]);
        }
}

//-------------------------------------------------------------
//      Observation Processing Loop
//-------------------------------------------------------------

for(iob=0; iob<nob; iob++){
     // Extract current observation.

     tob = time[iob];
     num = numdata[iob];
     iztype = datatype[iob];
     phase_angle = phase[iob];
     for(i=0; i<num; i++){
          z[i] = observation[i][iob];
     }

     //-----------------------------------------------------
     //     Propogate States and Phi
     //-----------------------------------------------------

     dynamics(phi, xpred, tob, modal_ref, phase_angle, num_sats,
              modal_pred);

     //  Store off states for plotting.

     for(i=0; i<size_state; i++){
          physical_states[iter-1][iob][i] = xpred[i];
     }
      for(i=0; i<size_state; i++){
          modal_states[iter-1][iob][i] = modal_pred[i];
```

```
}

//-------------------------------------------------------
//     Obtain Matrices for this Observation
//-------------------------------------------------------

obser_processor(tob, iztype, Qinv, zpred, h, xpred, num_sats);

//  Matrix Calculations - Current Observation.
//  Form the residual vector and test for rejection.

irej = 0;
for(i=0; i<num; i++){
     residuals[iter-1][iob][i] = z[i] - zpred[i];
     sqroot = sqrtl(Qinv[i][i]);
     if(fabsl(residuals[iter-1][iob][i]) > reject/sqroot){
          rej_flag[iter-1][iob][i] = 1;
          irej = 1;
     }
     else rej_flag[iter-1][iob][i] = 0;
}

//-------------------------------------------------------
//     Matrix Calculations
//-------------------------------------------------------
//  Check for observation rejection prior to calculations.
//  If rejected, skip calculations and obtain another
//  observation.

if(irej==0){

     //  Form matix product tmat=h*phi.

     for(i=0; i<num; i++){
```

```
     for(j=0; j<size_state; j++){
          tmat[i][j] = 0.00e+00;
          for(k=0; k<size_state; k++){
               tmat[i][j] = tmat[i][j] +
                            h[i][k]*phi[k][j];
          }
     }
}

//  Form matrix product T'*inv(Q).

 for(i=0; i<size_state; i++){
     for(j=0; j<num; j++){
          htq1[i][j] = 0.00e+00;
          for(k=0; k<num; k++){
               htq1[i][j] = htq1[i][j] +
                    tmat[k][i]*Qinv[k][j];
          }
     }
 }

//  Form product of T'*inv(Q)*T and progressively add.

for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          for(k=0; k<num; k++){
               pinv[i][j] = pinv[i][j] +
                    htq1[i][k]*tmat[k][j];
          }
     }
}

//  Form product T'*inv(Q)*r and progressively add.
```

```
            for(i=0; i<size_state; i++){
                for(j=0; j<num; j++){
                    htq1r[i] = htq1r[i] +
                                htq1[i][j]*residuals[iter-1][iob][j];
                }
            }
        }
    }

    //-----------------------------------------------------------
    //    Print first pass residuals
    //-----------------------------------------------------------

    if(iter==1){
        fprintf(out, "**  First Pass Residuals(in meters):  **\n");
        for(j=0; j<nob; j++){
            flag = 0;
            for(i=0; i<num; i++){
                //  6378.135 km is equal to 1 DU (mil-spec for
                //  earth radii.
                converted_resid = residuals[0][j][i]*6378135;
                fprintf(out, "%+.15Le " , converted_resid);
                if(rej_flag[0][j][i] == 1){
                    flag = 1;
                }
            }
            if(flag == 1){
                fprintf(out, "Rejected\n");
            }
            else fprintf(out, "\n");
        }
    }

    //-----------------------------------------------------------
```

```
//     Data is processed...improve estimate
//-------------------------------------------------------------
//  Invert pinv to find the covariance P.

//*****************************************************************
//  Debug code to print out covariance.
if((sample=fopen
  ("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\sample.out",
   "w"))==NULL){
     printf("Cannot open file.\n");
}
for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          fprintf(sample, "%+.15Le\n", pinv[i][j]);
     }
     fprintf(sample, "\n");
}
fclose(sample);
//*****************************************************************

matrix_inverter(pinv, p, ier, condnum, num_sats);
fprintf(out, "\n\n\n***Iteration %i Statistics***\n", iter);
fprintf(out, "\nCovariance matrix inversion error code = %i\n",
        ier);
fprintf(out, "\nCovariance matrix condition number = %.2Lf\n",
        condnum);

//  Multiply P by T'inv(Q)r to get corrections to states.

for(i=0; i<size_state; i++){
     diff_modal[iter-1][i] = 0.0e+00;
     for(j=0; j<size_state; j++){
          diff_modal[iter-1][i] = diff_modal[iter-1][i]
                                  + p[i][j]*htq1r[j];
```

```
    }
}

//  Check convergence.

converge = false;
for(i=0; i<size_state; i++){
    if(fabsl(diff_modal[iter-1][i]) <
        (0.01*(sqrtl(fabs(p[i][i]))))){
        converge = true;
    }
}

//  Print iteration.
fprintf(out, "\nIteration %i state corrections:\n", iter);
for(i=0; i<size_state; i++){
    fprintf(out, "%+.15Le\n", diff_modal[iter-1][i]);
}
fprintf(out, "\n");

cout << "Filter has completed " << iter << " iteration(s).\n";

//  Add in state corrections.

for(i=0; i<size_state; i++){
    modal_ref[i] = modal_ref[i] + diff_modal[iter-1][i];
}

//  Store modal states
for(i=0; i<size_state; i++){
    modal_ref_mat[iter][i] = modal_ref[i];
}

//  Print current best guess.
```

```
        fprintf(out, "\nCurrent state vector at epoch:\n\n");
        for(i=0; i<size_state; i++){
            fprintf(out, "%+.15Le\n", modal_ref[i]);
        }
        fprintf(out, "\n");

        //  Check to see if done.

        if(converge==true){
            finished = true;
        }
        if(iter==1){
            finished = false;
        }
        if(finished==true){
            fprintf(out, "\n\n\nCONVERGENCE ACHIEVED.\n\n\n");
        }

        //  If convergence achieved, break out of loop.

        max_iter = 0;
        if(finished == true){
            max_iter = iter - 1;
            iter = maxit + 1;
        }
    }
    //-----------------------------------------------------------------
    //     The EndGame
    //-----------------------------------------------------------------
    if(iob==maxit && finished==false){

        //---------------------------------------------------------
        //    Print Last pass residuals
```

```
        //-----------------------------------------------------------

        fprintf(out, "**  Last Pass Residuals(in meters): **\n");
        for(j=0; j<nob; j++){
            flag = 0;
            for(i=0; i<num; i++){
                //  6378.135 km is equal to 1 DU (mil-spec for
                //  earth radii.
                converted_resid = residuals[max_iter][j][i]*6378135;
                fprintf(out, "%+.15Le ", converted_resid);
                if(rej_flag[max_iter][j][i] == 1){
                    flag = 1;
                }
            }
            if(flag == 1){
                fprintf(out, "Rejected\n");
            }
            else fprintf(out, "\n");
        }
        fprintf(out, "\n\n\nConvergence not achieved.\n\n\n");
    }
    else
        //-----------------------------------------------------------
        //    Print Last pass residuals
        //-----------------------------------------------------------

        fprintf(out, "**  Last Pass Residuals(in meters):  **\n");
        for(j=0; j<nob; j++){
            flag = 0;
            for(i=0; i<num; i++){
                //  6378.135 km is equal to 1 DU (mil-spec for
                //  earth radii.
                converted_resid = residuals[max_iter][j][i]*6378135;
                fprintf(out, "%+.15Le ", converted_resid);
```

```
            if(rej_flag[max_iter][j][i] == 1){
                flag = 1;
            }
        }
        if(flag == 1){
            fprintf(out, "Rejected\n");
        }
        else fprintf(out, "\n");
}

//  Print covariance matrix.
fprintf(out, "\n\n\nCovariance Matrix at epoch is:\n");
for(i=0; i<size_state; i++){
        for(j=0; j<size_state; j++){
            fprintf(out, "%+.15Le\n", p[i][j]);
        }
        fprintf(out, "\n");
}

//  Print state at time of last observation.

fprintf(out, "\n\n\n***Last Observation Statistics***\n");
fprintf(out, "\nState at time t: %.15Le\n", tob);
for(j=0; j<size_state; j++){
        fprintf(out, "%+.15Le\n", xpred[j]);
}
fprintf(out, "\n");

//  Calculate covaiance at last ovservation time
//  via phi*p*phi'.  Use the last Phi calculated.

//  Calculate phi*p.

for(i=0; i<size_state; i++){
```

```
        for(j=0; j<size_state; j++){
            phip[i][j] = 0.00e+00;
            for(k=0; k<size_state; k++){
                phip[i][j] = phip[i][j] + phi[i][k]*p[k][j];
            }
        }
    }

    //  Calculate phi*p*phi'.

    for(i=0; i<size_state; i++){
        for(j=0; j<size_state; j++){
            p[i][j] = 0.00e+00;
            for(k=0; k<size_state; k++){
                p[i][j] = p[i][j] + phip[i][k]*phi[j][k];
            }
        }
    }

    fprintf(out, "\nCovariance at time of last ovservation:\n");
    for(i=0; i<size_state; i++){
        for(j=0; j<size_state; j++){
            fprintf(out, "%+.15Le\n", p[i][j]);
        }
        fprintf(out, "\n");
    }

    //  Output final time, stae estimate, pinv into file for input
    //  into Bayes filter.

    if((input=fopen
      ("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\state.new",
       "w"))==NULL){
        printf("Cannot open file.\n");
```

```
}

fprintf(input, "%.15Le\n\n", time[nob-1]);
for(i=0; i<size_state; i++){
     fprintf(input, "%+.15Le\n", xpred[i]);
}
fprintf(input, "\n");
fprintf(input, "%i\n\n", maxit);
fprintf(input, "%.15Le\n\n", reject);

//  Invert P.

matrix_inverter(p, pinv, ier, condnum, num_sats);

fprintf(out, "\nCovariance matrix inversion error code = %i\n",
        ier);
fprintf(out, "\nCovariance matrix condition number = %.2Lf\n",
        condnum);

fprintf(out, "\nCovariance inverse at last observation:\n");
for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          fprintf(out, "%+.15Le\n", pinv[i][j]);
     }
     fprintf(out, "\n");
}

for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          fprintf(input, "%+.15Le\n", pinv[i][j]);
     }
     fprintf(input, "\n");
}
```

```
//  Open files to output residuals, corrections, state
//  estimates, physical states and modal states for plotting.

if((type=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\type.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((meas=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\meas.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((phys=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\phys.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((modal=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\modal.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((resids=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\resids.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((resids_size=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\size.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((modal_corr=fopen
```

```
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\corrections.out",
"w"))==NULL){
printf("Cannot open file.\n");
}
if((modal_est=fopen
("c:\\windows\\desktop\\thesis\\working\\bayesfilter\\estimates.out",
"w"))==NULL){
printf("Cannot open file.\n");
}

//  Output residual array size [max_iter +1][nob][6]
//  Hard-coded to 6 resids.  With 3 sats, the max resids
//  per measurement will be 6 for DGPS data.

max = max_iter + 1;  //  Max_iter is zero based so add one.
fprintf(resids_size, "%i\n", max);
fprintf(resids_size, "%i\n\n", nob);

//  Output all residuals.

for(i=0; i<max; i++){
     for(j=0; j<nob; j++){
          for(k=0; k<6; k++){
               fprintf(resids, "%+.15Le ", residuals[i][j][k]);
          }
     fprintf(resids, "\n");
     }
     fprintf(resids, "\n\n\n");
}

for(i=0; i<max; i++){
     for(j=0; j<size_state; j++){
          fprintf(modal_corr, "%+.15Le ", diff_modal[i][j]);
     }
```

```
        fprintf(modal_corr, "\n");
    }

    for(i=0; i<max; i++){
        for(j=0; j<size_state; j++){
            fprintf(modal_est, "%+.15Le ", modal_ref_mat[i][j]);
        }
        fprintf(modal_est, "\n");
    }

    //  Output state estimates.
    // (currently hard-coded for last iteration)

    for(i=(max-1); i<max; i++){
        for(j=0; j<nob; j++){
            for(k=0; k<size_state; k++){
                fprintf(phys, "%+.15Le ", physical_states[i][j][k]);
            }
        fprintf(phys, "\n");
        }
        fprintf(phys, "\n\n\n");
    }
     for(i=(max-1); i<max; i++){
        for(j=0; j<nob; j++){
            for(k=0; k<size_state; k++){
                fprintf(modal, "%+.15Le ", modal_states[i][j][k]);
            }
        fprintf(modal, "\n");
        }
        fprintf(modal, "\n\n\n");
    }

    //  Output datatype and observation matricies.
```

```
for(i=0; i<nob; i++){
     for(j=0; j<9; j++){
          fprintf(meas, "%+.15Le ", observation[j][i]);
     }
     fprintf(meas, "\n");
}
for(i=0; i<nob; i++){
     fprintf(type, "%i\n", datatype[i]);
}

//  Prepare input file for next truthmodel data batch.
//  Currently optons for batch are hard-coded.

if((oneday=fopen
("c:\\windows\\desktop\\thesis\\fromwiesel\\truthmodel\\oneday.in",
"w"))==NULL){
printf("Cannot open file.\n");
}

egm_variable = 14;
tu = 108;
data_points = 100;
int_step = 500;
output_type = 1;
std_dev = 0;
seed = 1234567;
sps_gps_sigma = 0;
sectoral_flag = 0;
tesseral_flag = 0;

fprintf(oneday, "%i\n", egm_variable);
fprintf(oneday, "%.1d %i %i\n", tu, data_points, int_step);
fprintf(oneday, "%i %+.15Le %i\n", output_type, std_dev, seed);
fprintf(oneday, "%+.15Le\n", sps_gps_sigma);
```

```
            fprintf(oneday, "%i %i\n", sectoral_flag, tesseral_flag);
            fprintf(oneday, "%i\n", num_sats);
            k=0;
            for(i=0; i<(size_state/6); i++){
                 for(j=(0+k); j<(k+6); j++){
                      fprintf(oneday, "%+.15Le ", modal_ref[j]);
                 }
                 fprintf(oneday, "\n");
                 k=j;
            }

            //  Clean-up.

            fclose(type);
            fclose(meas);
            fclose(modal);
            fclose(phys);
            fclose(oneday);
            fclose(modal_corr);
            fclose(modal_est);
            fclose(resids_size);
            fclose(resids);
            fclose(out);
            fclose(input);
            //*********************************************************************
            //  Debug code to print out covariance.
            //fclose(sample);
            //*********************************************************************
            cout << "Program finished - press any key to continue...";
            getch();
        return 0;
}
//---------------------------------------------------------------------------
//---------------------------------------------------------------------------
```

```
//-------------------------------------------------------------------------
//     The "dynamics" routine.  It is passed t and modal_ref, and it gives
//     back the predicted state (x_pred) and the state transition matrix
//     (phi) at time t.
//-------------------------------------------------------------------------
//
//

void dynamics(long double phi[][18], long double xpred[],
              long double &tob, long double modal_ref[],
              long double &phase_angle, int &num_sats,
              long double modal_pred[])
{
     long double Xpo[6], expJt[6][6], F[6][6], phi_per_sat[6][6];
     long double propagate[18][18];
     int offset, factor, i, j, k, size_state;

     //  State size calculation.

     size_state = num_sats*6;

     //  Step one in bringing the state to time tob...
     //  Call soln1a to find the components necessary to build phi.
     //  These components are the periodic coefficient matrix, F, and the
     //  matrix exponential with J containing the Poincare exponents.

     soln1a(tob, phase_angle, Xpo, F, expJt);

     //  Generate phi for each sat.

     for(i=0; i<6; i++){
          for(j=0; j<6; j++){
               phi_per_sat[i][j] = 0.00e+00;
               for(k=0; k<6; k++){
```

```
                phi_per_sat[i][j] = phi_per_sat[i][j] + F[i][k]*expJt[k][j];
            }
        }
    }

    //  Form phi_ref based on num of sats by combining individual "phi's"

    for(i=0; i<size_state; i++){
        for(j=0; j<size_state; j++){
            phi[i][j] = 0.0e+00;
        }
    }
    offset = 0;
    factor = 1;
    for(k=0; k<num_sats; k++){
        for(i=(0+offset); i<(6*factor); i++){
            for(j=(0+offset); j<(6*factor); j++){
                phi[i][j] = phi_per_sat[i-offset][j-offset];
            }
        }
        offset = offset + 6;
        factor = factor + 1;
    }

    //  Multiply phi times moal_ref ----
    //  this gives physical states at tob (xpred_ref).

    for(i=0; i<size_state; i++){
        xpred[i] = 0.0e+00;
        for(j=0; j<size_state; j++){
            xpred[i] = xpred[i] + phi[i][j]*modal_ref[j];
        }
    }
```

```
    //  Multiply exp(Jt) times modal_ref to get modal state at time t.

    for(i=0; i<size_state; i++){
        for(j=0; j<size_state; j++){
            propagate[i][j] = 0.0e+00;
        }
    }
    offset = 0;
    factor = 1;
    for(k=0; k<num_sats; k++){
        for(i=(0+offset); i<(6*factor); i++){
            for(j=(0+offset); j<(6*factor); j++){
                propagate[i][j] = expJt[i-offset][j-offset];
            }
        }
        offset = offset + 6;
        factor = factor + 1;
    }

    for(i=0; i<size_state; i++){
        modal_pred[i] = 0.0e+00;
        for(j=0; j<size_state; j++){
            modal_pred[i] = modal_pred[i] + propagate[i][j]*modal_ref[j];
        }
    }
}

//-------------------------------------------------------------------------
//     The obser_processor routine  takes in t, data type, and x_ref, and it
//     gives back the predicted measurements (z_pred), H, and Q.
//-------------------------------------------------------------------------
//
//
```

```
void obser_processor(long double &tob_ref, int &iztype, long
double Qinv[][18],
                     long double zpred[], long double h[][18],
                     long double xpred[], int &num_sats)
{
     int i, j, size_state, sat_one, sat_two, flip_sign, m, k, sat;
     int factor, offset, loop;
     //  State size calculation.

     size_state = num_sats*6;

     //  The following code (1) calculates the predicted measurement and (2) H
     //  matirx depending on the data type.  Data type 1 is Carrier-Phase
     //  Differential Global Positioning (CDGPS) data.  Date type two is
     //  relative range data derived from clock pulses between satellites.

     if(iztype==1){
          k = 0;
          sat = 2;
          for(i=1; i<num_sats; i++){

               //  Initialize zpred to zero and then populate via G(x) for
               //  CDGPS data.  G(x) is position of ith satellite minus
               //  position of first satellite.

               for(j=0; j<3; j++){
                    zpred[k] = 0.0e+00;
                    zpred[k] = xpred[sat*3+j]-xpred[j];
                    k = k + 1;
               }
               sat = sat + 2;
          }

          sat = 1;
```

```
      for(k=0; k<(num_sats-1); k++){
          for(i=0; i<3; i++){

               //  Initialize H to zero.

               for(j=0; j<size_state; j++){
                    h[i+k*3][j] = 0.0e+00;
               }

               //  Form H given G(x) from above.

               h[i+k*3][i] = -1.0e+00;
               h[i+k*3][i+sat*6] = 1.0e+00;
          }
          sat = sat + 1;
     }

     //  Hard-coded Qinv.

      for(i=0; i<((num_sats - 1)*3); i++){
           for(j=0; j<((num_sats - 1)*3); j++){
                Qinv[i][j] = 0.0e+00;
           }
           Qinv[i][i] =  2.126045e+8; // 3 cm sigma
      }
}
else if(iztype==2){
     //  Initialize zpred data to zero.
     //  This algorithm requires 3 satellites

     for(i=0; i<((num_sats*num_sats-num_sats)/2); i++){
          zpred[i] = 0.0e+00;
     }
```

```
//  Populate zpred with estimated data given the G(x)
//  for relative range data.

sat_one = 0;
sat_two = 1;
k = 0;
for(i=1; i<num_sats; i++){
     for(j=i+1; j<(num_sats+1); j++){
          zpred[k] = sqrtl(
                     (xpred[sat_one*6] - xpred[sat_two*6])*
                     (xpred[sat_one*6] - xpred[sat_two*6])
                    +(xpred[sat_one*6+1] - xpred[sat_two*6+1])*
                     (xpred[sat_one*6+1] - xpred[sat_two*6+1])
                    +(xpred[sat_one*6+2] - xpred[sat_two*6+2])*
                     (xpred[sat_one*6+2] - xpred[sat_two*6+2])
                      );
           k = k + 1;
           sat_two = sat_two + 1;
     }
     sat_one = sat_one +1;
     sat_two = i + 1;
}

//  Initialize to H to zero.

for(i=0; i<((num_sats*num_sats-num_sats)/2); i++){
     for(j=0; j<size_state; j++){
          h[i][j] = 0.0e+00;
     }
}

//  Populate H matrix for range data given G(x) from above.

sat_one = 0;
```

```
        sat_two = 1;
        k = 0;

        for(i=1; i<num_sats; i++){
            for(j=(i+1); j<(num_sats+1); j++){
                h[k][sat_one*6] =
                    (xpred[sat_one*6] - xpred[sat_two*6])/zpred[k];
                h[k][sat_one*6+1] =
                    (xpred[sat_one*6+1] - xpred[sat_two*6+1])/zpred[k];
                h[k][sat_one*6+2] =
                    (xpred[sat_one*6+2] - xpred[sat_two*6+2])/zpred[k];
                h[k][sat_two*6] = -h[k][sat_one*6];
                h[k][sat_two*6+1] = -h[k][sat_one*6+1];
                h[k][sat_two*6+2] = -h[k][sat_one*6+2];
                k = k + 1;
                sat_two = sat_two + 1;
            }
            sat_one = sat_one +1;
            sat_two = i + 1;
        }

        //  Hard-coded Qinv.

        for(i=0; i<((num_sats*num_sats-num_sats)/2); i++){
            for(j=0; j<((num_sats*num_sats-num_sats)/2); j++){
                Qinv[i][j] = 0.0e+00;
            }
            Qinv[i][i] =  3.1890675e+8; // 2 cm sigma
        }
    }
    else{
        //  Absolute data relative to the periodic orbit. (i.e. GPS)
        for(i=0; i<(3*num_sats); i++){
            zpred[i] = 0.0e+00;
```

```
}

//  Populate zpred with estimated data given the G(x)
//  for relative range data.

offset = 0;
for(i=0; i<1; i++){   // set to num_sats for all
     for(j=0; j<3; j++){
          zpred[j + (offset/2)] = xpred[j + offset];
     }
     offset = offset + 6;
}

//  Initialize to H to zero.

for(i=0; i<(3*num_sats); i++){
     for(j=0; j<size_state; j++){
          h[i][j] = 0.0e+00;
     }
}

//  Populate H matrix for the absolute data given G(x) from above.
offset = 0;
for(i=0; i<1; i++){  // set to num_sats for all
     for(j=0; j<3; j++){
          h[j + (offset/2)][j + offset] = 1.0e+00;
     }
     offset = offset + 6;
}

//  Hard-coded Qinv.

for(i=0; i<(3*num_sats); i++){
     for(j=0; j<3*num_sats; j++){
```

```
                    Qinv[i][j] = 0.0e+00;
               }
               Qinv[i][i] =  2.126045e+6; // 3 m sigma
          }
     }
}

//-------------------------------------------------------------------------
//     Inverts a matrix via Guass eliminaiton with maximal pivoting.
//-------------------------------------------------------------------------
//
//

void matrix_inverter(long double a[][18], long double ainv[][18],
int &ier,
                     long double &condnum, int &num_sats)
{
     long double anorm, tol, amax, irr[18], var, x[18];
     long double row_norm_ainv, row_norm_a, row_sum;
     int i, j, id, ir, is, size_state;

     size_state = num_sats*6;

     //  Estimate condition number using row-sum norm (part 1).

     row_norm_a = 0.0e+00;
     for(i=0; i<size_state; i++){
          row_sum = 0.0e+00;
          for(j=0; j<size_state; j++){
               row_sum = row_sum + a[i][j];
          }
          if(row_sum > row_norm_a){
               row_norm_a = row_sum;
          }
```

```
}


// Initialize ainv as the identity matrix.

for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          ainv[i][j] = 0.0e+00;
     }
}
for(i=0; i<size_state; i++){
     ainv[i][i] = 1.0e+00;
}

//-------------------------------------------------------------------
//  Calculate Inverse
//-------------------------------------------------------------------
//  Find max norm of a.

anorm = 0.0e+00;
for(i=0; i<size_state; i++){
     for(j=0; j<size_state; j++){
          if(fabsl(a[i][j]) > anorm){
          anorm = fabsl(a[i][j]);
          }
     }
}

//  Set tolerance equal to 2^(-number_of_binary_digits_in_mantissa).

tol = 1.0e-12;
ier = 0;
id = 0;
for(i=0; i<size_state; i++){
```

```
        irr[i] = 999;
    }
    do{
        ir = 0;
        is = 0;
        amax = 0.0e+00;

        //  Find max pivot.

        for(i=0; i<size_state; i++){
            if(irr[i] == 999){
                for(j=0; j<size_state; j++){
                    var = fabsl(a[i][j]);
                    if((var - amax)>0){
                        ir = i;
                        is = j;
                        amax = var;
                    }
                }
            }
        }

        //  Singularity test.

        if(amax/anorm < tol){
            ier = 129;
        }

        //  Forward elimination.

        irr[ir] = is;
        for(i=0; i<size_state; i++){
            if(i != ir){
                var = a[i][is]/a[ir][is];
```

```
                for(j=0; j<size_state; j++){
                    a[i][j] = a[i][j] - var*a[ir][j];
                }
                a[i][is] = 0.0e+00;
                for(j=0; j<size_state; j++){
                    ainv[i][j] = ainv[i][j] - var*ainv[ir][j];
                }
            }
        }
        id = id +1;
}while(id<size_state);

//  Back subsitution.

for(j=0; j<size_state; j++){
    for(i=0; i<size_state; i++){
        ir = irr[i];
        x[ir] = ainv[i][j]/a[i][ir];
    }
    for(i=0; i<size_state; i++){
        ainv[i][j] = x[i];
    }
}

//  Estimate condition number using row-sum norm (part 2).

row_norm_ainv = 0.0e+00;
for(i=0; i<size_state; i++){
    row_sum = 0.0e+00;
    for(j=0; j<size_state; j++){
        row_sum = row_sum + ainv[i][j];
    }
    if(row_sum > row_norm_ainv){
        row_norm_ainv = row_sum;
```

```
        }
    }
    condnum = row_norm_a*row_norm_ainv;

}

//-------------------------------------------------------------------------
//  SOLN1A - Based on Floquet theory, calculates components necessary
//  for propogating state.
//-------------------------------------------------------------------------
//
//
void soln1a(long double &t, long double &PHI, long double Xpo[],
            long double F[][6], long double expJt[][6])
{
        long double ckXpo[6][30], skXpo[6][30], tau, Omegaz;
        long double reckF[6][6][30], reskF[6][6][30];
        long double Omega[2], sine[30], cosine[30], zz, OrbitW;
        long double imckF[6][6][30], imskF[6][6][30];

        ifstream in;

        int ifirst, maxcoef, npts, i, j, k;

        ifirst = 0;
        maxcoef = 30;

        //  Read in datafile if this tis the first call.

        if(ifirst==0){
             ifirst = 1;
             in.open(
             "c:\\windows\\desktop\\thesis\\fromwiesel\\truthmodel\\order1.dat",
             ios::in);
```

```
if(!in.is_open()){
     cout << "The input file could not be opened.\n";
}

//  Orbital Period, nodal regression frame rate.

in >> tau >> Omegaz;
in >> npts;

//  Check for storage space.

if(npts > maxcoef){
     cout << "soln1:  Not enough storage for fourier coefs.\n";
     // code to exit out
}

//  The one non-degenerate pair of poincare exponents.

in >> Omega[0];
in >> Omega[1];

//  Read in the periodic orgit fourier coefficients.

for(i=0; i<6; i++){
     in >> ckXpo[i][0];
     for(j=0; j<npts; j++){
          in >> ckXpo[i][j+i] >> skXpo[i][j];
     }
}

//  Read in fourier coefficients of orbital Floquet modal matrix.

for(j=0; j<6; j++){
```

```
            for(i=0; i<6; i++){

                //  Real part of coefficients.

                in >> reckF[i][j][0];
                for(k=0; k<npts; k++){
                    in >> reckF[i][j][k+1] >> reskF[i][j][k];
                }
            }
        }

        in.close();

}

//  Evaluate sine & cosine buffers.

//  Calculate Orbital Frequency.

OrbitW = 2.0e+00*3.141592653589e+00/tau;
for(i=0; i<npts; i++){
    sine[i] = sinl((i+1.0)*(OrbitW*t + PHI));
    cosine[i] = cosl((i+1.0)*(OrbitW*t + PHI));
}

//  Sum the Fourier series.

for(i=0; i<6; i++){

    //  The periodic orbit.

    Xpo[i] = ckXpo[i][0];
    for(j=0 ;j<npts; j++){
        Xpo[i] = Xpo[i] + ckXpo[i][j+1]*cosine[j]
```

```
                    + skXpo[i][j]*sine[j];
        }
        for(j=0; j<6; j++){

            //  The F matrix.

            zz = reckF[i][j][0];
            for(k=0; k<npts; k++){
                zz = zz + reckF[i][j][k+1]*cosine[k]
                        + reskF[i][j][k]*sine[k];
            }
            F[i][j] = zz;
        }
    }

    //  Evaluate exp(Jt) from the two Poincare exponents and the structure
    //  of the other modes.

    for(i=0; i<6; i++){
        for(j=0; j<6; j++){
            expJt[i][j] = 0.0e+00;
        }
    }

    //  The non-degenerate mode.

    expJt[0][0] = cosl( Omega[0]*t );
    expJt[0][1] = sinl( Omega[0]*t );
    expJt[1][1] = cosl( Omega[1]*t );
    expJt[1][0] = sinl( Omega[1]*t );

    //  The degenerate energy / angular momentum nodes.

    expJt[2][2] = 1.0e+00;
```

```
        expJt[3][3] = 1.0e+00;
        expJt[4][4] = 1.0e+00;
        expJt[5][5] = 1.0e+00;

        expJt[2][3] = t;
        expJt[4][5] = t;

}

//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
```

Bibliography

1. Binning, Patrick W. "A Widelane Integer Resolution Technique for Relative Navigation between Spacecraft." *Proceedings of the 11th International Technical Meeting of the Satellite Division of the Institute of Navigation*. 1577–1586. 1998.

2. Brouwer, Dirk and Gerald M. Clemence. *Methods of Celestial Mechanics*. New York: Academic Press, 1961.

3. Cantafio, Leopold J. *Space-Based Radar Handbook*. Norwood, MA: Artech House, Inc., 1989.

4. Carpenter, J. Russell and Emil R. Schiesser. "The Importance of Semi-Major Axis Knowledge in the Determination of Near-Circular Orbits," *Advances in Astronautical Sciences*, *102*:1297–1310 (1999).

5. Chiaradia, Ana P. M. and others. "The Use of the Single Frequency GPS Measurements To Determine In Real Time Artificial Satellite Orbits." *IAF, International Astronautical Congress, 50th, Amsterdam, Netherlands*. 449–459. 1999.

6. Corazzini, Tobe and Jonathan P. How. "Onboard GPS Signal Augmentation for Spacecraft Formation Flying." *Proceedings of the 11th International Technical Meeting of the Satellite Division of the Institute of Navigation, Nashville, TN*. 1937–1946. 1998.

7. Cox, Duncan B. and John D. W. Brading. "Integration of LAMBA Ambiguity Resolution with Kalman Filter for Relative Navigation of Spacecraft." *Proceedings of the ION 55th Annual Meeting*. 739–745. 1999.

8. Davis, Capt Jeffery. *Navigation of Satellite Clusters*. MS thesis, AFIT/GA/ENY/00M-01, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2000 (AD-A1103970).

9. Filer, Capt Sherrie N. *Investigation of the Observability of a Satellite Cluster in a Near Circular Orbit*. MS thesis, AFIT/GA/ENY/89D-2, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, December 1989 (AD-A1103970).

10. Greenwood, Donald T. *Classical Dynamics*. Mineola: Dover Publications, Inc., 1977.

11. Hacker, T.L. and R.J. Sedwick. "Space-Based GMTI Radar Using Separated Spacecraft Interferometry," *AIAA Space Technology Conference and Exposition, Albuquerque, NM* (1999).

12. Johnston, Capt Stephen C. *Autonomous Navigation of a Satellite Cluster*. MS thesis, AFIT/GA/ENY/90D-9, Graduate School of Engineering, Air Force Insti-

tute of Technology (AETC), Wright-Patterson AFB OH, December 1990 (AD-A1103970).

13. Kaplan, Elliot D. *Understanding GPS Principles and Applications.* Boston: Artech House Publishers, 1996.

14. Kiraly, Zsolt and others. "The Orion Project: A Space Formation Flying Experiment." *Proceedings of the AIAA Space Technology Conference and Exposition.* 1999.

15. Kong, Edmund M. C. and others. "Exploiting Orbital Dynamics for Aperture Synthesis Using Distributed Satellite Systems: Applications to a Visible Earth Imager System," *Advances in the Astronautical Sciences*, *102*:285–301 (1999).

16. Larson, Wiley J. and James R. Wertz. *Space Mission Analysis and Design.* Torrance, California: Microcosm, Inc. and Kluwer Academic Publishers, 1992.

17. Maybeck, Peter S. *Stochastic Models, Estimating, and Control I.* New York: Academic Press, 1982.

18. Micheal D. Violet, Principal Systems Engineer, Space Systems Group. Telephone interview. Orbital Sciences Corporation, Herndon, VA, 12 February 2001.

19. Psiaki, Mark L. "Autonomous Orbit Determination For Two Spacecraft From Relative Position Measurements." *AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Boston, MA.* 1998.

20. Raquet, Maj John. Class handout distributed in EENG 633, Advanced GPS Theory and Applications.Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, June 2000.

21. Raquet, Maj John. Class handout distributed in EENG 533, Navigation Using GPS.Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2000.

22. Reid, J. Gary. *Linear System Fundamentals. Continuous and Discrete, Classic and Modern.* New York: McGraw-Hill, 1983.

23. Sabol, Chris and others. "Satellite Formation Flying Design and Evolution." *AAS/AIAA Space Flight Mechanics Meeting.* 1–20. San Diego, CA: AAS Publications Office, 1999.

24. Schaub, H. and K. Alfriend. "J2 Invariant Relative Orbits for Spacecraft Formations," *NASA Flight Mechanics Symposium, NASA* (1999).

25. Sedwick, R. J. and others. "Mitigation of Differential Perturbations in Clusters of Formation Flying Satellites," *Advances in the Astronautical Sciences*, *102*:323–342 (1999).

26. Sedwick, R.J. and others. "Optimum Aperture Placement for a Space-Based Radar System Using Separated Spacecraft Interferometry," *AIAA Guidance, Navigation, and Control and Exhibit, Portland, OR* (1999).

27. Shaw, G. B. and others. "Generalized Characteristics of Communication, Sensing, and Navigation Satellite Systems," *Journal of Spacecraft and Rockets*, *37*:801–811 (2000).

28. Sparks, Anthony. "Satellite Formationkeeping Control in the Presence of Gravity Perturbations." *Proceedings of the American Control Conference*. 844–848. 2000.

29. Ward, Capt Michael L. P. *Estimated Satellite Cluster Elements in Near Circular Orbit*. MS thesis, AFIT/GA/AA/88D-13, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, December 1988 (AD-A1103970).

30. Wie, Bong. *Space Vehicle Dynamics and Control*. Reston, VA: American Institute of Aeronautics and Astronautics, Inc., 1998.

31. Wiesel, William. *Spaceflight Dynamics*. Boston: McGraw-Hill, 1989.

32. Wiesel, William. Class handout distributed in MECH 636, Advanced Astrodynamics.Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 1998.

33. Wiesel, William. Class handout distributed in MECH 731, Modern Methods of Orbit Determination.Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, June 1998.

34. Wiesel, William E. "Optimal Impulsive Control of Relative Satellite Motion." Submitted to Journal of Guidance and Control, November 2000.

35. Wiesel, William E. "Perturbations of Relative Satellite Motion about an Oblate Planet." Submitted to Journal of Guidance and Control, June 2000.

36. Wiesel, William E. "Relative Satellite Motion about an Oblate Planet." Submitted to Journal of Guidance and Control, January 2000.

Vita

Captain Ralph E. Bordner III was born in Peoria, Illinois. He was raised in Vero Beach, Florida from the age of 10, and he graduated from the Vero Beach Senior High School in 1991. He graduated from the University of Central Florida in Orlando, Florida in 1996 with a Bachelor of Science in Aerospace Engineering and a Minor in Aerospace Studies. After commissioning on 5 May 1996, he reported to the 2nd Space Operations Squadron at Schriever AFB in Colorado Springs, Colorado. There he served as a Satellite Vehicle Officer, Spacecraft Systems Analyst, and Operations Requirements Analyst for the Global Positioning System. He departed Colorado Springs in August of 1999 to attend the School of Engineering at the Air Force Institute of Technology. Upon graduation he will be assigned to the National Air Intelligence Center at Wright-Patterson AFB, Ohio.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
08-03-2001	Thesis	Mar 2000 - Mar 2001

4. TITLE AND SUBTITLE
ESTIMATION OF RELATIVE SATELLITE FORMATION ELEMENT POSITIONS IN NEAR CIRCULAR ORBITS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Ralph E. Bordner, Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
Graduate School of Engineering and Management
2920 P Street, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GA/ENY/01M-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFRL/VAAD
Dr. Andrew Sparks
2130 8th Street
WPAFB OH 45433

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Precise relative position determination of satellite formation elements in near circular orbits is the focus of this thesis. A Bayes filter is used for the estimator with the dynamics solution obtained by solving the time periodic, linearized system via Floquet theory. This approach is similar to that of the traditional Clohessy-Wiltshire treatment, however much more of the system's behavior is retained by using a linear, time periodic system. The dynamics model includes all zonal harmonics of the Earth as well as sectoral, tesseral, and air drag perturbations. The filter algorithm was fed three types of simulated measurements: relative Carrier-Phases Differential GPS data, Standard Positioning System GPS data and range measurements between each satellite. Simulations were conducted to investigate the accuracy obtainable with these measurements. Results obtainted are on the order of the noise of the measurements, which is as low as 2 centimeters in each coordinate axis.

15. SUBJECT TERMS
Satellite Clusters, Relative Satellite Navigation, Estimation, Satellite Formation Flying

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. William E. Wiesel Jr., AFIT/ENY
U	U	U	UU	133	19b. TELEPHONE NUMBER *(Include area code)* (937)255-6565 x4312